Phase I Final Report

SPACEBORNE MEMORY ORGANIZATION

by

D. C. Gunderson
C. W. Hastings
G. J. Prom

April 1966

Prepared Under Contract No. NAS 12-38 by

HONEYWELL INC.
SYSTEMS AND RESEARCH DIVISION
Minneapolis, Minnesota

Electronics Research Center
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

ABSTRACT

The application of associative memories to the data processing problems expected to be encountered on future space vehicles has been studied. Four problem areas considered in detail are (1) data acquisition and compression, (2) picture data processing, (3) incremented computations, and (4) miscellaneous control and memory functions present in multi-processor systems. An evaluation of the associative memory capabilities required for these and other applications was performed, and two associative memory organizational approaches are identified as being the most useful for space applications. A device rating system was also devised, and the devices suitable for each of four associative memory organizational approaches are rated.

CONTENTS

CONTENTS

ILLUSTRATIONS

TABLES

SECTION I
INTRODUCTION

This is the final report of Phase 1 on Contract NAS 38-12, Spaceborne Memory Organization. This study has been aimed at the application of associative memories to the computational problems expected to be encountered on future space exploration missions. Associative memories with their search and processing capabilities are found to be applicable in a number of significant problem areas.

This study consisted of five major tasks as shown in Figure 1.

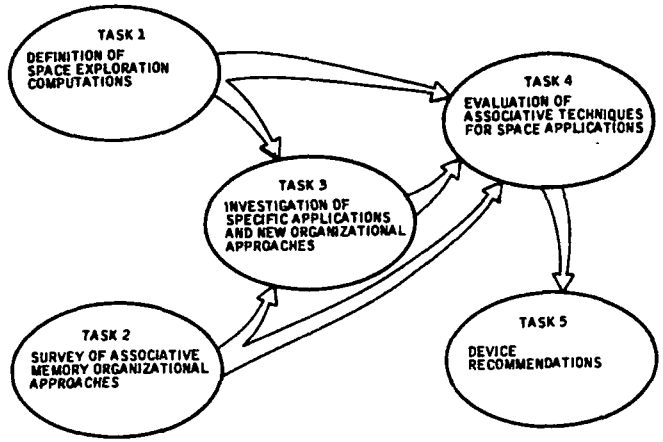


Figure 1. Study Organization

The results of Task 1, Task 2, and a portion of Task 3 were previously reported in an interim report (Ref. 1) dated 15 December 1965. In Task 1, the definition of the space exploration computation requirements, a number of specific problem areas were identified as having characteristics such that associative memories can be advantageously applied.

Task 2, a survey of associative memory organizational approaches, resulted in the definition of five basic approaches which cover the range from a minimum capability memory to a very powerful one. Each of the approaches is described in Reference 1 in terms of its searching, processing, reading and writing capabilities, and in terms of the devices suitable for its mechanization. An organizational approach strong on processing operations was defined as an associative processor. All the applications investigated make use of one or more of these five organizational approaches. An extensive device survey was also carried out as part of Task 2.

In Task 3, a number of the applications identified in Task 1 were investigated in some detail. These are:

- An associative processor for data acquisition
- Some applications of associative memories in multiprocessor systems
- An associative processor for picture processing
- An associative processor for incremental computation

These applications are described in detail in Section II of this report.

The results of Task 4, the evaluation of associative memories for space applications, are described in Section III. The applications listed above, plus a number of others, which were either investigated outside of this study or in much less detail than those listed above, are summarized. Some general conclusions regarding the types of associative memories that will be useful on space vehicles are given.

In Task 5, device recommendations were made for each of those organizational approaches identified in Task 4. A scheme for computing figures of merit for suitable components for each of the selected organizational approaches and a discussion of specific device recommendations is given in Section IV.

Section V is a summary of the results and recommendations of the entire study.

SECTION II
APPLICATION INVESTIGATIONS

As a part of Task 1 of the study, a number of computational problems were identified as potential applications of associative memories. Four of these selected for detailed investigation are as follows:

- An associative memory to provide an adaptive data acquisition capability
- Associative memories for control and memory functions in a multi-processor system
- An associative memory for incremental computations such as are required in navigation and control functions
- An associative memory for certain picture processing tasks.

The results of these investigations are described in this section.

ASSOCIATIVE PROCESSOR FOR DATA ACQUISITION

This subsection describes a fundamentally new scheme for a data acquisition system: the use of an associative processor to simultaneously monitor the values of many different sensors and to select for permanent retention data from whatever sensors are currently exhibiting the most "significant" activity. According to this scheme, the associative computer functions both as a "data compressor" and as a "multiplexer".

The approach taken to the task of "data compression" is quite unlike that taken in existing space vehicle data systems, in that the outputs of all sensors are

examined jointly to determine which is instantaneously the most "significant" rather than basing a decision regarding the "significance" of each sensor reading only on the time history of that particular sensor. The "data compression" procedures which follow from this approach may be quite analogous to those appropriate for use with a single sensor, but any attempt to analyze or simulate the system mathematically would now entail analyzing or simulating the operation of the entire system rather than that of just one sensor, in order that the model would realistically portray the behavior of these procedures in non-trivial cases.

The essential contribution of the associative computer in the system is to provide a straightforward mechanization of the ability to consider, simultaneously, quantities calculated from the data for all sensors in making a decision as to which sensor or sensors should currently be taken as having the most significant data. The resulting procedure is similar in some respects to what has been called "adaptive sampling", but the latter implies (see Reference 2, pages 2-2 and 2-3) the adaptive alteration of some fixed sequence of sensor sampling, whereas there just is no fixed sequence in the approach to be described here.

This work is described primarily in terms of potential applications to tasks aboard a long-range exploratory aerospace vehicle, because it was directed at such application in accordance with the contract which supported it. However, the techniques and type of device described could be of some utility in almost any type of data acquisition system wherever the sheer amount of data to be handled is large enough to motivate the use of some editing procedure.

Data Compression Philosophy

The terms "data compression" and "data editing" properly refer to any systematic technique for extracting, from a large number of sensor readings, the "most significant" information. The objective is to reduce the total amount of

information which must be retained, or transmitted, to as close as possible to the minimum which will represent the time history of the sensor variables to a desired accuracy. There are a very large number of these techniques (see Reference 2, Part 2). The only ones described here are those which involve a relatively simple adaptive decision-making process, based on the data themselves or on simple quantities readily calculated from the data.

Why Data Compression is Becoming Attractive -- The amount of rocket fuel required to put a satellite or space vehicle outside the earth's gravitational field is very great, and its cost is obviously minimized by keeping the weight of the vehicle as low as possible. In the past, this consideration has led to aerospace computers being designed as extremely rudimentary devices, lacking many of the programming conveniences taken for granted even in low-cost ground-based computers. The advent of integrated circuits, however, changed the economics of aerospace computer design to a degree where such an extreme approach is no longer necessary; the most recently designed aerospace computers, such as the Alert computer made by Honeywell Aeronautical Division at St. Petersburg, Florida, now have many of the worthwhile programming features long familiar to users of ground-based scientific and real-time computers. Integrated circuits have a very high physical packing density, consume very little power, and improve the reliability picture over what it was with discrete semiconductors. Substantial, although less spectacular, progress has also been made in the field of aerospace digital memories. All in all, it is no longer prohibitively expensive to put an appreciable amount of computing power aboard an aerospace vehicle.

At the same time that the economic penalty is decreasing for the inclusion of very powerful computing devices on board an aerospace vehicle, the penalty is increasing with respect to the amount of data which can be transmitted from the vehicle back to ground during a mission. One reason is that the power required to transmit data from a vehicle back to earth is considerable in terms of the power supplies available. Moreover, as the vehicle ventures further from

the earth, still more power is required to produce a given level of signal-to-noise ratio at the receiver on earth, and this problem will be vastly aggravated on interplanetary and interstellar missions. The second major reason is that the number of sensors included aboard vehicles is subject to rapid increase as more and more experiments are thought up and more and more previously ignored factors are taken into consideration. The sheer amount of data being transmitted is already such as to stagger not only the transmission capabilities of the satellites (let alone the imagination), but also the capabilities of the computation centers on earth (at, for instance, NASA-Goddard and JPL) to record, "decommutate"*, and analyze the data. NASA-Goddard, for instance, has to analyze about a trillion such pieces of data per year, or roughly 30,000 per second.

This data traffic jam would be more easily tolerated if all of the data were truly necessary. However, much of it is "redundant" in the sense that the time history of the variables of interest can be known adequately without it. A given sensor may exhibit very little change for a period of days or months; although the fact that the sensor is quiescent is necessary information, sending a great number of unchanged sample readings is an inefficient way of communicating this fact. Also, there may be reasons why the outputs of some sensors are not felt to be important during certain phases of a mission.

To sum up, the time is approaching when it will be feasible to put very powerful computing systems aboard relatively small space vehicles, but it will not be feasible for these space vehicles to transmit back to earth more than a minute portion of the data gathered by the vehicle's sensors, and it will not even be economical to require that the receiving stations and data processing centers on earth cope with the full amount of data which could be gathered by the vehicle's sensors. Hence, both the capability which can be put aboard a vehicle to do data compression and the need for this capability will be subject to very rapid increase as integrated circuits get smaller and smaller and space missions get longer and longer.

*Sensor data is "commutated" into a complicated sequential format for transmission. The process of unscrambling this format so that the data for each given sensor are all together, with a post-reckoned approximate time of sampling for each datum, is referred to as "decommutation".

Matching the Amount of Message to the Transmitter Capability -- Because this discussion will use some terms in a slightly different way than they have been previously used, some definitions are in order. Terms underlined in this subsection are considered to be defined by their context in an obvious way.

First of all, the general type of procedure described will not be referred to as "adaptive sampling", because the sampling of a sensor is considered here to consist of the conversion of its output into digital form and the recording of this digital value in a memory, and this operation is now completed for all sensors on every sampling cycle. Rather, the purpose of the "adaptive" procedures is to decide what samples are sufficiently worthy of further attention that they should be transmitted from the vehicle to the receiving station. Hence, what is going on is really adaptive editing and not "adaptive sampling".

To permit the consideration of slightly more general categories of data-originating devices, such devices will be referred to hereafter as data sources rather than as "sensors". A sensor is still one possible type of data source. The term data source is here understood to imply a scientific instrument whose output is of value either for research purposes or for vehicle and system control computations.

Data compression techniques now in general use examine the time history of one source and make a decision as to what samples from that source are to be transmitted strictly on the basis of this time history without considering the relative importance of this source as compared to others. Procedures of this type will be referred to as single-source techniques. Single-source techniques have by now been investigated extensively; Reference 2 is a very comprehensive report on them. Single-source techniques will be contrasted in this discussion with all-sources-jointly techniques, which do take the relative importance of all data sources into consideration; procedures of this type have received comparatively little previous study.

Not only is there always a severe premium on transmitter power consumption, but the capability of the vehicle to transmit data may vary with time. If the vehicle encounters an environmental disturbance such as intense heat or radiation, its electronic system may incur intermittent or temporary malfunctions which lead to an increase in the number of transmitter errors. The data being transmitted are usually encoded in an error-detection (or possibly an error-detection-and-correction) format; if the number of errors becomes unreasonably large, the vehicle may have to retransmit data already transmitted and may even slow down its rate of transmission either to incorporate greater coding redundancy in the message or to use some more reliable transmission mode. Consequently, the data transmitter can best be treated as having a time-varying transmission capability, which the electronic master control apparatus aboard the vehicle will know at any given time, but which may vary over a considerable range.

Because of the possible need for retransmission of data already sent, and because the rate of the data transmitter must be matched since it is generally a synchronized device, most schemes for complete data compression systems based on single-source techniques presuppose a data buffer, which is a digital memory into which data selected for transmission is stored and from which data is read out for transmission. In normal system operation, this buffer is maintained partially full to a certain operating point. If the incoming data rate exceeds the transmitting data rate for a sufficient length of time, buffer overflow becomes possible, this condition is highly undesirable, as some data will be lost, and the identity of the lost data will be determined by chance rather than by the priorities implemented in the data compression system. Reference 2 contains a very detailed analysis of "queuing control algorithms" for the prevention of buffer overflow, totaling 175 pages -- which is some indication of the importance of the problem.

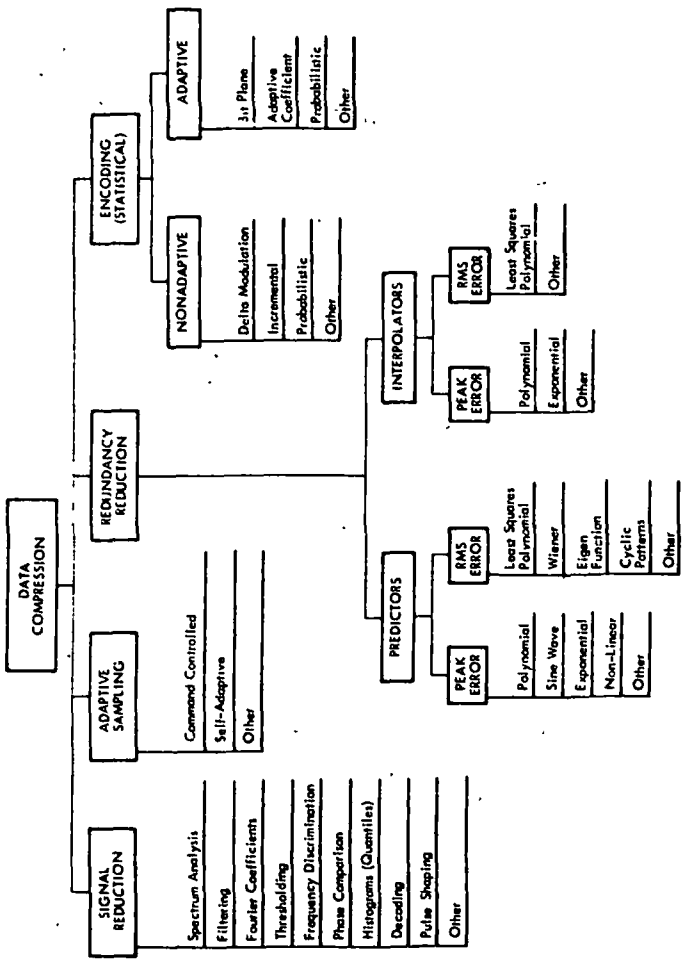
One consequence of the use of all-sources-jointly techniques is that the danger of buffer overflow disappears because it is now possible to adjust the rate of selection of data for transmission directly to the instantaneous transmission

capability; thus, only as much data will be selected for transmission within any given period of time as there is a reasonably good probability of being able to send. A buffer memory may still be useful to synchronize the data rate to the transmission rate; but there would be no need to include a large enough buffer to handle any possible worst case or to use any additional queuing control technique besides that of controlling the number of data read out from the associative memory on each sampling cycle -- unless the system were designed to allow for the outright failure of the transmitter for an extended period of time.

Single-Source Techniques -- One possible classification scheme for data compression techniques is shown in Figure 2, which has been taken from Reference 2. All of the techniques shown are single-source techniques with the exception of those listed under "Adaptive Sampling". Almost all of them could reasonably be implemented for a multi-source data acquisition system using an associative processor to good advantage. Some of those listed under "Signal Reduction" have previously been examined at Honeywell in other portions of the work on this contract.

The techniques with which this discussion are concerned are those listed under "Interpolators" -- in particular, "Peak Error Polynomial Interpolators". Reference 2 selects six methods from those in the categories shown in Figure 2 as generally superior; of these six, four were in the category of "Peak Error Polynomial Interpolators", one of the others was a "Peak Error Polynomial Predictor", and the remaining one was an "Adaptive Bit Plane" method under the heading of "Encoding".

In general, all of these methods share the advantage that they perform satisfactorily with "non-stationary" signals, whereas some of the other



12017-FR1

Figure 2. Classification of Data Compression Models (From Reference 2)

methods misbehave badly if the signal is "non-stationary".* A stationary signal is one whose limiting statistical parameters do not vary with time; a non-stationary signal is one not subject to this restriction. To state this definition another way, a stationary signal is one which, if processed extensively, will yield empirical statistical parameters which, except for normally distributed sampling error fluctuations, behave the same, regardless of the time one commences observing the signal; a non-stationary signal is one which yields quite different empirical statistical parameters if observed beginning at different times. Signals from real sensors are typically quite non-stationary.

The term "polynomial" is somewhat misleading, in that the only "polynomial" schemes seriously considered in Reference 2 use either "zero-order polynomials" (constant segments) or "first-order polynomials" (linear segments of non-zero slope). No practical advantage was found by the authors of Reference 2 in going to higher-order polynomial representations of signal time histories.

The essential idea of both "predictors" and "interpolators" is to examine the data points observed over a period of time and to send only a single piece of data as representative of a sequence of several data points (the "zero-order" case); or else to send two parameters** which determine a line having a given slope, and passing through a given point, such that the line is a "fit" to a sequence of several data points (the "first-order" case*). The difference between "predictors" and "interpolators" lies in the nature of the computations performed and the choice of data points used at each step. A predictor first fits a horizontal or sloped line (or in the higher-order cases, a polynomial curve) to a number of preceding data points; each new data point observed is

*There are several mathematically sophisticated methods, for instance, which require knowledge of the signal spectrum in order to be able to calculate expected values of various statistics. Not only is the spectrum unlikely to be "known" in an exploratory mission, but these techniques may also lead, in effect to, "filtering out" unexpected values of sensor data, which is obviously undesirable behavior in any procedure intended for use in an exploratory research vehicle.

**These will typically both be points, but one could be a slope and the other could be a point if preferable.

12017-FR1

checked to see if it is within some tolerance ϵ of the value predicted from the fitted curve. In the case of interpolators, each new point observed is, in effect, incorporated along with recent previously observed points into a data sequence to which a curve is fitted; and all of the points of the current data sequence are tested to see if they are within ϵ of the value given by the fitted curve.

The phrase "in effect" is important; for most zero-order and first-order interpolators, it is not actually necessary to store an indeterminate number of data points and test them one by one; this circumstance is fortunate, for otherwise the use of interpolators would carry with it the penalty of unpredictable fluctuations in the need for "scratch pad" storage for sample points. The trick is to use a "corridor-type" algorithm; this term is used here since the terminology for the same thing is "dual prediction" algorithm in Reference 2 which is very confusing, since this type of algorithm is used with interpolators rather than with predictors. The operation of several corridor-type interpolator algorithms, and also of a few predictor algorithms, is described in some detail in Appendix A of this report.

The Use of an Associative Processor

The single-source techniques discussed in Appendix A could be used in a data acquisition system exactly as they are presented there if a suitable arithmetic computation and comparison device were available for each data source. A general-purpose computer equipped with an input multiplexer could also do the job; it would simply input each data point as it was sampled and would maintain a separate area of memory for the data and other information for each data source. However, the general-purpose computer is inherently a sequential device and would have to handle each data point one after another; since the application here is basically one of handling many data points from as many data sources in parallel, the computer would have to simulate parallelism by operating in the usual one-after-another manner at very high speed and using some form of indexing.

If the number of data sources is large and the sampling rate for each is moderate to high, it becomes expedient to consider types of parallelized computing devices whose logical "shape" more nearly matches that of the job to be done; otherwise for larger and larger systems a more and more high-speed general-purpose computer and multiplexer would be required, and that solution to the problem ultimately becomes rather expensive. An associative processor which is defined as an associative memory having facilities for performing all standard arithmetic operations in each word, is one attractive type of parallelized device. Arithmetic operations in an associative processor are not usually ultra-high-speed since addition and subtraction are performed serially and multiplication and division entail iterated addition and subtraction. However, since as many arithmetic operations can be carried out at a time as there are words in the associative processor and since the indexing operations needed by the general-purpose computer to do parallelized jobs are dispensed with, the effective useful arithmetic operations-per-second rate of an associative processor can be very high.

An associative processor retains the basic abilities of an associative memory to do searches. In particular, equality search, inequality search, maximum/minimum search, and certain other search operations can readily be implemented by use of the subtraction capability, even though they might be performed in a rather different manner if the associative memory were not also required to function as an associative processor. Thus the question arises if there is not some useful application to which these search capabilities can be put, once an associative processor has been chosen as the central element of the data acquisition system by reason of its parallelism. It turns out that the search capabilities are precisely the feature which allows the associative processor to simultaneously consider quantities calculated from the data received by all data sources, and hence to efficiently determine the relative significance of the instantaneous readings from these sources. Thus, although it is quite feasible to use an associative processor to implement single-source techniques for data compression separately for each source, it is equally feasible now to switch to all-sources-jointly techniques, which would be quite difficult to implement with most other hardware configurations.

All-Sources-Jointly Techniques -- Almost any imaginable criterion for deciding when to transmit the sample from a single source will involve a preset allowable tolerance ϵ and some "discrepancy" Δ which measures the instantaneous actual tolerance of the data. In the case of single-source techniques, a decision is made to transmit whenever the instantaneous actual tolerance exceeds the preset tolerance. In the case where some large number (such as 64 or 256) of sources are being simultaneously monitored by an associative processor, this rule-of-thumb is no longer the only possible one, although it could still be applied, see for example the description, presented in Reference 3, of a device which - despite the name "associative data compressor" - is basically intended to efficiently apply single-source techniques to up to 4096 data sources at once using a mixture of analog and digital processing.

To state the criterion for the single-source case another way, if the discrepancy for the i^{th} data source is indicated by Δ_i, and the corresponding preset tolerance by ϵ_i, the criterion for transmission is that the absolute value of the quotient Δ_i / ϵ_i is greater than one. In the all-sources-jointly approach, on the other hand, the i^{th} quotient is stored in a pre-assigned field in the i^{th} word of the associative processor, "maximum searches" are made for the largest numbers in this field of any word, and the data selected for transmission are those corresponding to the largest tolerance quotients.

It may be seen that the preset tolerance ϵ_i assigned to the i^{th} data source is inversely related to the priority which is in effect given to that data source. In actual operation, it would be preferable to store $1/\epsilon_i$ in each word rather than ϵ_i, thus entailing multiplication operations in the associative processor rather than divisions. If another priority factor, P_i, were multiplied by the tolerance quotient, the effect would be that Δ_i was multiplied in turn by two preset constants, which would be time-consuming and unwarranted.

Organization of a Data Acquisition System -- The basic plan of the adaptive data acquisition system as now conceived is shown in Figure 3. The major subsystems are the associative processor with some extra hardware for

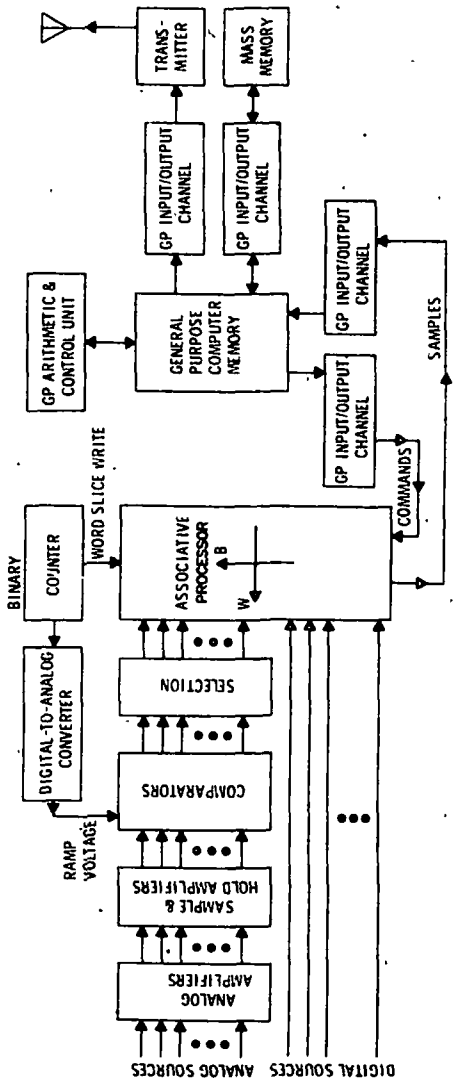


Figure 3. Adaptive Data Acquisition System

analog-to-digital conversion, a general purpose computer connected to the associative processor in a "symbiotic"arrangement, the transmitter, and, optionally, a mass memory device which would probably be an aerospace-environment magnetic tape recorder.

The associative processor would be of the type 2a organization in the classification scheme developed by Honeywell and presented in the midterm report on this contract (Reference 1) for reasons discussed later. The general-purpose computer would probably be the system master control computer as well. It relies on the associative processor to cope with what would otherwise be, for a general-purpose computer, an overwhelmingly severe problem of converting, multiplexing, and compressing data; and it in turn serves the associative processor as an instruction memory, hence the use of the term "symbiotic". The memory of the general-purpose computer can be allocated, if need be, to provide a small amount of buffer storage for data received from the associative processor for forwarding to the transmitter as fast as the latter can send it.

If the system is required to be able to continue acquiring data during a period of complete transmitter shutdown and to retain this data for transmission whenever the transmitter can again operate, there is obvious need for a memory of larger capacity than the random-access memory of the general-purpose computer or the specialized memory of the associative processor. Small, ruggedized digital magnetic tape recorders have been developed for aerospace applications by several vendors, and one of these would be suitable. However, if a satisfactory solid-state mass memory with no moving parts could be developed to withstand an aerospace environment, it would of course be preferable.

Decentralized A-D Conversion

Some of the instruments which would be used aboard aerospace exploratory vehicles are inherently digital; high-energy particle counters, timer counters, laser gyroscopes, and others fall into this class. Some instruments may present their reading basically as a physical displacement or shaft rotation; the electrical output from these may be either digital or analog at will, since a photoelectric or magnetic Gray-code wheel will present a digital output, and a slide-wire resistor will present an analog output. Many instruments, however, are intrinsically analog devices; thermocouples, Hall-effect magnetic field detection probes, light-intensity measuring cells, rotor gyroscopes, piezoelectric pressure and strain indicators, and various types of apparatus for analyzing soil and atmospheric samples all most naturally present an analog reading.

In a data acquisition system using an associative processor it would be awkward system design to multiplex many analog signals, convert them to digital through the usual sort of centralized successive-approximation or subranging analog-to-digital converter (see Reference 4, and digitally demultiplex them so that each signal was again matched one-for-one with a word of associative processor. Fortunately, there are quite straightforward ways of parallelizing analog-to-digital conversion, one appropriate to the design situation considered here is discussed in the following paragraphs.

The essential principle of decentralized analog-to-digital conversion is to generate a central voltage reference signal, whose digital equivalent is continuously available, and compare this reference signal with each incoming analog signal, using a separate comparator for each incoming signal. Where there is no reason to suppose that certain analog values are the only ones ever assumed, a convenient way to generate this reference voltage and its digital equivalent is to use a binary counter, whose value is converted to a voltage by a digital-to-analog converter. If the counter counts at a constant rate, the voltage produced will be a "ramp" function of time, possibly with slight stair-steps if the response of the digital-to-analog converter is sufficiently fast.

So that only a single time marker need be stored for all data sampled on one sampling cycle, all data sources are sampled simultaneously, and the analog readings are stored in "sample-and-hold" amplifiers. Such an amplifier may be thought of as a capacitor to store a voltage plus a switch which may connect the capacitor either to the source or to the reading circuit (here a comparator), although the actual circuit would be more complex than this.

Also, it is customary to pass all input analog signals through some sort of amplifier for isolation and scaling so that the analog signals as received by the comparators all have the same "offset" (minimum) and the same "full-scale" (maximum) values. Here, the offset and full-scale values for each data source would be standardized as the analog equivalents of the zero and full-scale values of the binary counter. In a conventional data acquisition system, the reason for the scaling is to maintain maximum accuracy during the process of analog multiplexing and analog-to-digital conversion; here, there is the additional consideration of using the individual comparators to best advantage.

The system just described is illustrated in the upper left-hand portion of Figure 3. Its operation proceeds as follows: On each sampling cycle, the binary counter starts from zero and counts at an even rate up to full-scale (for a 10-bit counter, full-scale is $2^{10} - 1 = 1023$). A comparator is provided for each associative processor word corresponding to an analog data source, and one terminal of the comparator is connected to the ramp voltage and the other to the source via its sample-and-hold amplifier, all incoming source voltages are simultaneously available to the appropriate comparators. When the ramp voltage first exceeds the incoming source voltage for a comparator, the comparator changes state and provides a logical output which "selects" the corresponding processor word; the current value of the binary counter is then immediately recorded into that word, using the "word slice writing" technique described in Reference 1. One advantage of this technique is that the counter value can, if necessary, be written simultaneously into a number

words if it happens that a number of data sources have equal voltages thin the minimum binary resolution of the counter) on a given sampling cle. It may be seen that, when the counter reaches full scale, all analog a source readings will have been converted to their digital equivalents h the latter stored in the appropriate processor words.

stem Status Monitoring -- The task of system status monitoring resembles t of adaptive data acquisition in the nature of the operations performed, hough its purpose is completely different in function within the over-all oloratory vehicle system. The various control devices, power sources, l other pieces of equipment aboard a vehicle will typically be subject to note control from earth through the system master control computer obably a general-purpose computer). The ground station will learn the ndition of each device by way of a status word of binary information, intained by the device and available to be read by the control computer. ch bit of a status word normally indicates that some particular condition present or not present; in some cases, a status word may have a field of veral bits whose contents are interpreted as a numerical value. There is hard and fast line between what is status information and what is sensor ormation; in general, it may be assumed for present purposes that the tus information is whichever data is not of interest in connection with the entific investigation that the vehicle is carrying out, but is needed in ler that the control system can keep the vehicle and its subsystems operating perly.

e criteria for deciding when to transmit status information back to earth substantially, although not completely, different from the corresponding teria in the case of scientific information. It may be desired to keep a rly complete record of the status of various vehicle subsystems in order benefit future missions, but this activity is probably less important than t of keeping a complete record of the data from the vehicle's sensors obtained the course of probing an unknown environment. Status information must, vever, be noticed and relayed promptly back to earth in the event of any

potentially serious fault conditions for which corrective action must be taken. It may therefore be arranged that potential danger signals can be recognized by encoding them as numerical status words and having the associative processor search them on each sampling cycle to see if any is "out of tolerance" -- that is, over a given threshold. Here, however, "out of tolerance" means in a danger zone; and the rule for transmitting a status condition must be to transmit any that exceed the preset tolerance threshold, rather than transmitting only the few which are furthest out of tolerance. Because this rule would lead to at least a remote possibility of overloading the transmitter, and because danger signals should have priority even over scientific information, a likely rule-of-thumb would be to subtract the number of status words which must be sent on a given sampling cycle from the total number of words which can be sent for that cycle, leaving as the difference the number of sensor readings which can be sent on that cycle. If this rule is to be a useful one, the tolerances for status words must be carefully chosen so that they will very rarely be exceeded, and hence it will infrequently be necessary to send a lot of status information; otherwise, the amount of "administrative" information might reach the point of interfering seriously with the ability of the transmitter to send an adequate amount of scientific data back to ground.

It may be advisable, in a practical aerospace exploratory vehicle control system, to maintain two levels of tolerance for status words: one where the condition is serious enough to be called to the attention of the master control computer and another where the condition is much more serious and must be called to the attention of the earth-based control station. The system would probably be arranged so that the master control computer would take care of detecting any condition serious enough to exceed the second tolerance and of accordingly notifying the control station so that the associative processor would need only to see that the master computer received all status words which exceeded the first tolerances.

ower level of priority, it would also be desirable periodically to
n status words from all vehicle subsystems and transmit these to
., as a continuing record of the performance of each subsystem. Such
is not the primary purpose of an exploratory mission, but is needed in
· that vehicles for later missions be improved in performance and
bility.

m Organization

·e 3 gives the system organization in block diagram form. A brief
ssion follows on why this organization seems appropriate, including
he associative processor should have the "type 2a" organization of
·ence 1.

sing a Type of Associative Device -- A large number of approaches to
iative memory and associative processors are summarized in Table 3.1
ference 1. Basically, they fall into five major categories: minimum
iative memory, associative memory with bit-slice writing, associative
ory with all-parallel ("local logic") equality search, associative memory
ternary output from each bit storage cell, and associative memory
intercommunicating bit storage cells. Machines falling into the first
ategories can be constructed with any good non-destructive readout
ent suitable for a word-organized or "linear select" random-access
ory. Machines falling in the third and fourth categories can also be
ructed with any such element, subject to the additional proviso that
eadout signal from the element must be sufficiently accurate and uniform
signal cancellation logic techniques can be employed. Machines falling
e fifth category must generally be constructed entirely from active elements
as integrated circuit flipflops and logic circuits. The first, second, and
categories are in turn subdivided according to whether certain additional
bilities are provided. Particular subcategories correspond to: the use of
-information or "read-only" storage elements, the reduction of external

(presumably active logic element) bit storage from two bits per word to one, the addition of a serial adder circuit per computer word, and the use of cyclic-access rather than random-access type storage elements.

There are of course many different organizational approaches which could be used for an associative processor intended for adaptive data acquisition. However, the best choice intuitively is the one which includes all the capabilities necessary but is not slanted toward capabilities which are not important, unless of course the associative processor used for adaptive data acquisition is also to be used for some diverse purposes which require these capabilities. It will be assumed here than the only other task for which this same associative processor will be used is that of "system status monitoring", which is very similar to adaptive data acquisition in the required type of operation.

In the first place, the associative processor must be able to perform certain "field arithmetic" operations in order to handle the computations required for interpolation-type data compression procedures; that is, every word in the associative processor must be capable of adding two of its fields together and storing the results in a third field ("field addition") and similarly for subtraction and for multiplication. The capability of field division must also be included if first-order interpolator algorithms are to be used. There is even some possibility that the capability for floating-point operations may have to be included; these operations can be performed in an associative processor having certain additional features. The floating-point data format has advantages in a data acquisition system whenever variables of extremely large range are considered; however, the range problem could most likely be handled in the present system context simply by increasing the word length and staying with fixed-point arithmetic in the associative processor. After the data are read from the associative processor into the general-purpose computer, through which they would pass before being transmitted in any case, they could then be converted to a floating-point format if any message length reduction could be achieved thereby. Arithmetic operations will clearly be a major portion of the total processing load of the associative processor.

There is also a substantial requirement for the inequality and maximum (minimum) search operations. On every sampling cycle, the ratios Δ_i / ϵ_i of discrepancies to preset tolerances must be searched for the n largest such quotient values, where n is the number of data words which the transmitter can send to earth during the following sampling cycle. In the case of system status monitoring, all status words which have exceeded the preset danger point must be detected by a search. There may also be occasion, in the system status monitoring application, to search for certain logical combinations of bits; but it may be assumed here that this type of search is not absolutely necessary, since these logical combinations can always be encoded as a binary word in such a way that forbidden combinations correspond to large binary numbers. In any case, equality search will be a relatively unimportant operation relative to inequality and maximum searches, and no need whatever is anticipated for proximity search, therefore, there is no reason to consider the three more complex types (3, 4, and 5 in Reference 1) of associative memories which make use of "local logic".

It is evident that at least most of the bits in the associative processor word would need to be electrically alterable; even preset tolerances are likely to change during the course of a long mission. There is only one situation where the same datum stored in a word will be read from a great many times before being changed; this type of operation occurs when reading data from the associative processor into the master computer, since the "steering tag" - the code number identifying a particular data source, and stored in the word corresponding to that source -- never changes. Non-destructive readout memory elements would be desirable for storing preset tolerances and steering tags but would need to be electrically alterable for the tolerances. The greatest convenience would, therefore, ensue either from the choice of a highly reliable destructive readout element, or from the choice of a non-destructive readout element with a relatively fast write time. However, different types of elements could be used in different "field-slices" of the complete memory. (A "field-slice" comprises a number of adjacent bit slices, corresponding to all bit positions of a data field.)

Even though destructive-readout memories can be considered for the associative processor proper, it should be noted that the program storage memory for the associative processor should, if at all possible, be non-destructive read-out since the program will be read out over and over again hundreds of millions of times without alteration.

The type of associative processor organization considered here would therefore be either approach 2a (serial-adder-per-word) or approach 2c (cyclic-access), with the stipulation that a cyclic-access memory would also have a serial adder per word. However, because of the need for multiplication operations, the cyclic-access element would have definite latency problems, and random-access elements are therefore preferable. The origin of these latency problems is that, in multiplication, the field of an associative processor word which holds the multiplicand must be read out repeatedly; and, unless a separately readable cyclic-access element were provided just to contain that field of the word and not the whole word, a great deal of time would be wasted waiting for a complete cyclic timing period to occur so that readout might begin again. It would also be rather inconvenient to read out the multiplier bit-by-bit from a memory composed of cyclic-access elements. There is, however, some possibility of using cyclic-access elements of the asynchronous initiation, asynchronous read-write type (such as magnetic domain-wall shift registers); devices of this class can have multiple "read stations" if necessary, to permit reading from several fields of a word at the same time. Nevertheless, the cleanest solution would appear to be the choice of organizational type 2a, based on a random-access memory element suitable for non-destructive read-out operation but capable of fast writing.

The number of sensors in a typical exploratory space vehicle would probably be of the order of a few hundred one Honeywell estimate (Reference 5) was that a "maximum" would be 882. Adding to this total perhaps half as many status conditions to be monitored, it would appear that an associative computer of 512, 1024 or 2048 words would be as large as would usually need to be considered for this application. As may be seen from Table 3.1 of Reference 1,

the general category of memory elements which fit this size requirement is that of hysteresis elements; the memory is somewhat too large to be economically plausible at present on the basis of integrated circuits, particularly since there is absolutely no need for their "local logic" capabilities. Closed-flux magnetic thin-film elements (that is, plated-wire elements) are typical of the class of hysteresis elements capable of non-destructive readout operation and will, therefore, be taken here as a representative element on which to base the design of an actual device. For the purpose of estimating computing rates, it may be assumed that plated wire elements can be read from at the rate of 3 megacycles and written at a rate of 1 megacycle; these figures are relatively modest compared to many claims made for elements in a comparable stage of development, since rates considerable faster than these have already been achieved with plated-wire elements on the basis of laboratory experimentation.

The Rest of the System -- It may appear unusual to include two complete arithmetic computers in a system like this, but the economics of the system strongly favor doing so. Their capabilities nicely complement each other for this application.

First, a system master control computer on board the vehicle is probably necessary anyway, and would alsmost certainly need to be general-purpose in nature because of the variety of operations to be performed.

Second, no other provision has been made for an instruction memory for the associative processor besides using a portion of the memory of the general-purpose computer, an instruction memory is necessary, and should be capable of feeding the associative processor commands as fast as they can be executed, probably in a parallel word format. Moreover, no other provision has been made for a buffer storage device between the associative processor and the transmitter; here again, the use of a portion of the memory of the general-purpose computer is eminently plausible. Hence, if a general-purpose computer were not included, there would have to be some extra

random-access memory modules anyway. Since the major cost, size, weight, and power consumption of present-day general-purpose computers is that of the memory and not that of the arithmetic and control unit, a relatively small increment in hardware is involved in making these memory modules into a general-purpose computer.

Third, once this arrangement has been instituted, it lends itself very conveniently to adaptive modification of the preset tolerances in the associative processor by the general-purpose computer. Such modifications may be needed to preserve data integrity during periods of sensor quiescence (see Reference 2), and also to prevent transmitter overload during periods of extreme sensor activity. Also, the general-purpose computer can control the number n of data points to be retrieved, for transmission, from the associative processor by means of search operations after each sampling cycle.

Fourth, in many applications there is reason to suppose that the transmitter may be unable to transmit for periods of a few minutes, a few hours, or a few days -- for instance, the vehicle may be behind another celestial body with respect to the earth or caught in a radiation storm which temporarily provides so much background noise that transmission may as well be discontinued. If the system design philosophy is that the system is expected to be able to go on collecting data anyway and to transmit this data back to earth when a favorable opportunity comes, a mass memory is also necessary; and the memory of a general-purpose computer is again a good buffer for communicating with a mass memory.

As may be seen from Figure 3, the general-purpose computer needs a total of four input/output channels each of which would operate in an independent, asynchronous, memory-cycle-stealing mode. The arithmetic and control unit of the general-purpose computer would not have to attend to them except to periodically reset the "current address" for each channel, and perhaps once in a while to change the "limit address" of the channel also. During

normal operation of such a channel, the current address specifies the next main memory word to be input or output on that channel, and is counted up by 1 after each word is handled; the limit adress corresponds to the last word of the block of words in main memory to be handled this way. When the current address reaches the limit address, the input or output process ceases, and an interrupt signal is sent to the arithmetic and control unit.

ME APPLICATIONS FOR ASSOCIATIVE MEMORIES IN MULTI-PROCESSOR STEMS

While the previous applications in this report have dealt with processing requirements of one kind or another, this application is concerned with control and memory functions in a multi-processor system. It is conceivable that associative memories would be used also as processors in the system to be described, but such applications will not be considered here.

General Description of a Spaceborne Multi-Processor

Before describing the general organization of the spaceborne multiprocessor, it is well to review the objectives of a spaceborne computer system. The three most important requirements are:

- Reliability -- A spaceborne multi-processor must be ultra-reliable. A single component failure can not be allowed to affect appreciably the computing power of the system.
- Flexibility -- The system must in some cases be capable of serving many users or programs simultaneously and in other situations must be able to apply the bulk of its computing power to a single high-priority program.

- Expandability -- It is desirable that the system be made up of a set of building blocks which can be combined in varying numbers to obtain computer systems of various sizes.

To meet the objectives outlined above for a spaceborne application, a multi-processor system with multi-program and parallel processing capabilities is proposed. Each of these features is described in terms of the objectives of the system:

- Multi-Processor -- The processing or computing capability of the system is provided in modularized form. Instead of one large processor, a number of smaller processors are used to provide an equivalent amount of computing capability. Such modularity helps meet the reliability requirements since loss of a single module will only degrade the system. It also allows various size systems to be realized by varying the number of processors used.

- Multi-Programming -- The multi-program feature gives the system the capability of simultaneously satisfying the requirements of many users.

- Parallel Processing -- This feature allows several of the processors of the system to be working simultaneously on the same program, the number depends on the amount of parallism in the program. Thus, a high-priority program can obtain a large portion of the computing power of the system.

A functional block diagram of the proposed multi-processor system is shown in Figure 4. The processors of the system are of two types -- task processors and I/O processors. The task processors, which need not necessarily be identical, handle all instructions to be performed except input/output instructions, which are handled by I/O processors. To allow parallel

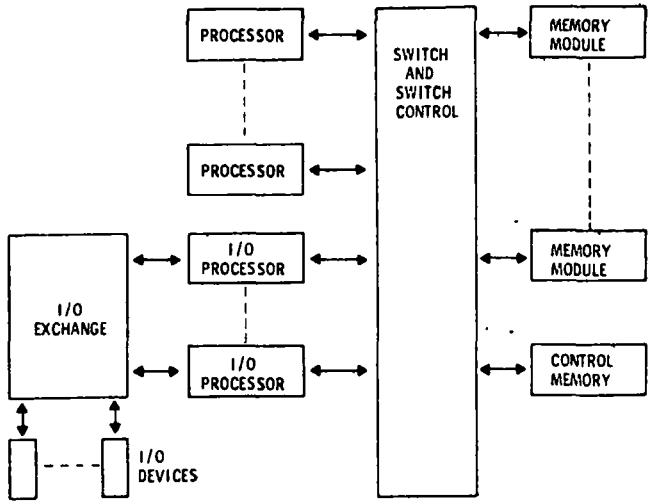


Figure 4. Functional Block Diagram of Multi-Processor

processing, programs must be stored in a central location so that all processors can be frequently reassigned. Thus, processors have no appreciable amount of memory of their own but make use of a bank of memory modules through use of a centralized switch.

An I/O exchange is shown in Figure 4 to interconnect I/O processors and I/O devices. This is necessary so that the failure of a single I/O processor does not permanently disconnect a user from the system.

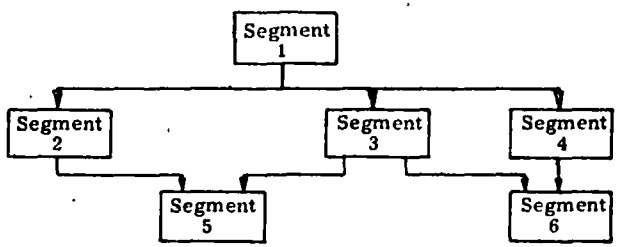
The control functions in this system are relatively complex because it is a multi-processor and has multi-program and parallel processing capabilities. Many of these functions are assumed to be performed by hardware rather than by the usual executive program. The Control Memory is included for this purpose. An associative mechanization of the Control Memory and the details of the functions it performs are discussed in the following subsection.

An Associative Memory for Control Functions

The main function of the Control Memory is to handle the assignment of tasks to processors, including I/O processors. However, other functions such as memory allocation are also included.

Task Assignment -- The programs in this multi-processor system are divided into segments on the basis of parallel paths. This is necessary in order to do parallel processing, which is defined as the capability of having more than one processor work simultaneously on a given program. The program segments are assigned to an available processor whenever they are ready to be performed.

The method proposed for handling the task assignment function can best be explained by considering the following program:



Each of the above segments is a sequence of instructions to be done by a single processor. Each segment is described by a state word which contains the information required by a processor to perform the segment. Assume that all six state words for the above program are initially stored in the Control Memory, although for large programs this may not be desirable. The state word has the following contents:

Status	Segment Identity	Segment Description

The status field indicates whether or not the segment is ready to be performed; the segment identity field contains such items as program name, segment name, etc.; and the segment description field includes such items as starting address, contents of index registers, etc. Initially, only the state word for segment 1 will have a status field indicating "ready" status. Thus, a processor searching for an assignment will select that state word, assuming that it is the highest priority program that has "ready" segments. When segment 1 is completed, the state words of segments 2, 3, and 4 can be set to "ready" status. These segments represent parallel paths in the program, and they can be performed simultaneously if three processors are available. When segment 2 is completed, the state word for segment 5 must be altered, but not necessarily to "ready" status since segment 3 must also be completed before segment 5 is ready. The same situation exists for segment 6. Thus, it can be seen that the number of processors working on a program is dependent on the parallel paths in the program and the number of processors available.

The status field of the state word, over which processors search to obtain an assignment, contains not only ready bits and the priority of the program, but also a time tag that indicates when the state word became ready A processor will select that state word with the highest priority that has been ready the longest. Thus queuing of requests is accomplished.

Tasks for I/O processors are selected by using the same general techniques. This points out the need for the search including that part of the "segment identity" field which specifies the type of segment (I/O or general processing).

Another aspect of the task assignment function is that of external interrupts. It is proposed that the users (either subsystems or personnel) make their requests for computations by writing state words into the Control Memory or changing the status of state words permanently stored there. Since the need for external interrupts usually arises either because inputs must be read immediately to prevent their being lost or because a processing task must be completed within a certain time, it appears that processors will never have to be interrupted in this system. This is true because a number of words of memory can be permanently assigned to critical inputs, and because processors frequently search over state words to find high priority tasks. Note that frequent searches for a new assignment by each processor can be assured by limiting the length of a program segment.

The Control Memory requirement can be nicely satisfied by an associative memory. The requirement can be described as table look-up or information retrieval from a file, the contents of which is frequently changing. Also, the retrieval operation must be performed over various fields of the file, thus making it difficult to maintain an ordered file and use conventional table look-up procedures. An associative memory, however, since its searches are performed simultaneously over all words, can perform rapid retrievals from a completely unordered file.

ıe specific operations to be performed by the Control Memory include the uality search, the inequality search, and the maximum (minimum) search. ıe state word selection function requires the intersection of several of the ove search operations. Since many of the functions to be performed consist simply of changing the status of state words by searching over the segent identity field, the equality search will be used more frequently than e others. This, plus the fact that many processors are sharing the Control emory, makes an all-parallel equality search appear justifiable. While no ocessing operations are anticipated, it appears that the bit slice writing pability would be useful to change the status of many state words simulneously. A word select ladder also would be useful to locate available ords in the Control Memory one at a time.

ıe size of the Control Memory will vary from system to system. However, nerally it falls into the "small" category previously defined to be less than 00 words.

emory Allocation -- In such a system it is assumed that the main memory not large enough to hold all programs simultaneously. Therefore, programs d data are stored in a secondary storage device and are brought into main emory when required. Since the mix of programs in main memory at any e time cannot be pre-specified, dynamic allocation of storage is required. ıis capability of running a program from anywhere in memory also necestates an address translation either when the program is brought into main emory or when it is run.

ıe details of this memory allocation function can vary extensively from stem to system. Therefore, the discussion will be concluded simply by serving that the requirements are much like those of task assignment, ıs it is likely that a single associative memory could be used to handle th functions.

Look-Aside Associative Memory

Another application for associative memories in a computer system is aimed at the reduction of the number of accesses to main memory. It can be seen that in a multi-processor system with a bank of shared memory modules this is particularly important. The solution is to place a look-aside associative memory (Ref. 6) at each processor.

The memory is used in the following way: Each instruction and operand, after it is used, is stored in the associative memory, rather than simply being discarded, in anticipation that it may be needed again in the near future. It can be recognized that this would be the case if a loop was present in the program. Also, all results of operations are stored in the associative memory rather than in the main memory address specified. This is done because it is assumed that in most cases the results produced are intermediate rather than final results. In these situations, two accesses to main memory are saved -- one to store the intermediate result and one to fetch it again. One of the difficult problems encountered in using this memory consists of deciding which quantity should be discarded. Each time a quantity is written into the associative memory, one must be removed, since the associative memory is always filled, except at the very start. In general, the one that should be removed is the one that has remained unused in the memory for the longest time. Note that if the word selected for removal is an instruction, it is simply erased since another copy exists in main memory. However, if it is a result, it must now be written into main memory.

The format of the word in the look-aside associative memory is as follows:

Main Memory Address	Instruction, Operand or Result	Status

Searches are performed over the address portion and possibly over the status portion depending on the method of handling it.

The look-aside associative memory must have the capability of doing an equality search in a time which is short compared to the main memory cycle time. Thus, the all-parallel equality search is a necessity. No other search or processing capabilities are required. There is also no need for a bit-slice writing capability or for a word select ladder.

The Organization of An Ultra-Reliable Multi-Processor

While the multi-processor system described in the previous subsection has many of the desirable characteristics for spaceborne applications, there are some undesirable characteristics also. The centralized switch interconnecting processors and memory modules, as well as the memory modules themselves, are not particularly strong features from the reliability point of view. Methods of reorganizing the memory and the switching are considered in this section. An ultra-reliable multi-processor system based on these multi-access memories is then described.

A Multi-Access Memory -- One way of reorganizing the memory and the switch to allow simultaneous access to the memory for a number of processors is shown in Figure 5.

The memory in this system is made multi-access by simply rotating the stored information 90 degrees from the way it is stored in a conventional random access memory. Thus, all words can be read out simultaneously, each in a serial-by-bit mode. The memory can be simultaneously accessed by many processors by providing each with one gate for each word of the memory. The contents of the memory is presented in a synchronous cyclic manner, either by shifting or by strobing a bit slice at a time, to each of the columns labeled "output switch" in Figure 5, and each processor, during each cycle, can select any word desired by closing the appropriate gate.

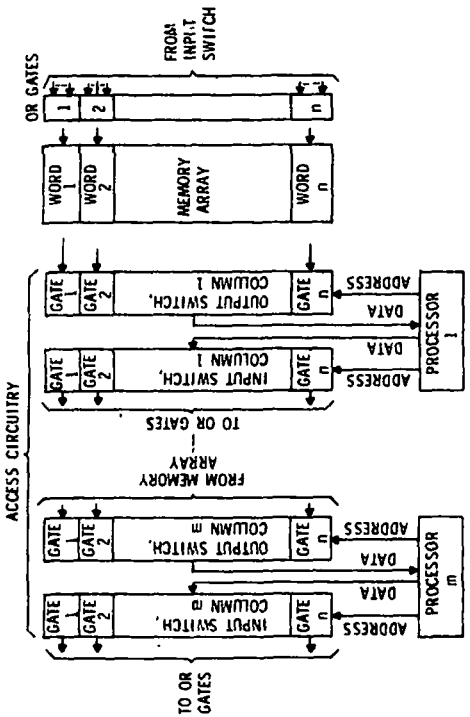


Figure 5. Location-Addressable Multi-Access Memory

'riting is accomplished by using the input switch which is organized similarly
› the output switch. The outputs of the gates along a row of the input switch
-e OR'ed together. The output of each OR gate in turn serves as the input
› a write driver.

it is assumed that the memory array is continuously cycling, each processor
an be either reading or writing each cycle. If desired, two or more processors
ould be reading the same word at the same time.

he advantages of this organization are primarily in the area of reliability.
nce each processor has its own switch, a failure in the switch will affect
ily one processor. The method of organization of the memory also provides
possible reliability advantage. Failure of a sense amplifier, for instance,
ill affect only one word rather than a single bit of all words as in the conven-
onal memory organization.

he most obvious disadvantage to this approach is cost. The number of gates
. the two switches is twice the number of words of memory times the number
processors involved. Thus, for 8,192 words and 16 processors, a total of
$18 = 262,144$ gates are required. In addition, each processor must, in this
xample, decode a 13-bit address to select one of 8,192 gates. To allow
odularity in the number of words of memory, the switch and the address
ecoder would probably be organized in modules. For instance 2^8 gates and
e decoder necessary to select one of 2^8 would be put in a single module.
he first five bits of a 13-bit address would then be used to select one of 32
odules, and the eight least significant bits would be sent to the address de-
oder in the selected module. While the hardware required to mechanize the
ates and address decoders is significant, it is felt that the advanced inte-
rated circuit technology will make this approach and variations of it de-
cribed below feasible.

n Associative Multi-Access Memory -- For relatively little additional cost,
ie multi-access memory described above can be made associative (content

addressable). This amounts to storing an address in each word of memory, distributing the address of the desired word to each gate, and letting the gate control itself rather than sending individual gate control signals to each gate.

The only change required to make the location-addressable, multi-access memory of Figure 5 into an associative multi-access memory is to supplement each gate of the switch array with a search capability. This is done as shown in Figure 6. One way of looking at the effect of this change is that the address (search word) can be submitted to the switch array serially rather than in parallel as in the location addressable case.

The search is performed by initially setting the results register to 1 and then comparing in a serial-by-bit mode each stored address with the address in the search register. Whenever a mismatch occurs in any word, the corresponding F/F is cleared to 0. Assuming only one word satisfies the search, only one F/F of the results register will contain a 1 when the search is completed. Since the contents of the results register directly controls the gates of the output switch, the selected word can now be read serially into the I/O register.

Note that the search and the read operation can be performed simultaneously by all processors. While it can be expected that processors would normally be reading different words, two or more processors could select and read the same word at the same time if desired.

The writing capability is included by providing one column of "input switch" for each processor. This set of gates is also controlled by the results register when a write operation is desired. The input switches of all processors are OR'ed, with the output of the OR gate serving as the input to a write driver. The capabilities described above can be provided with only five leads interconnecting processor and array. These are write enable, compare enable, set results register, data for writing and searching, and data for reading. While these facilities do provide basic search capabilities, additional hardware would be required to perform searches involving multiple matches.

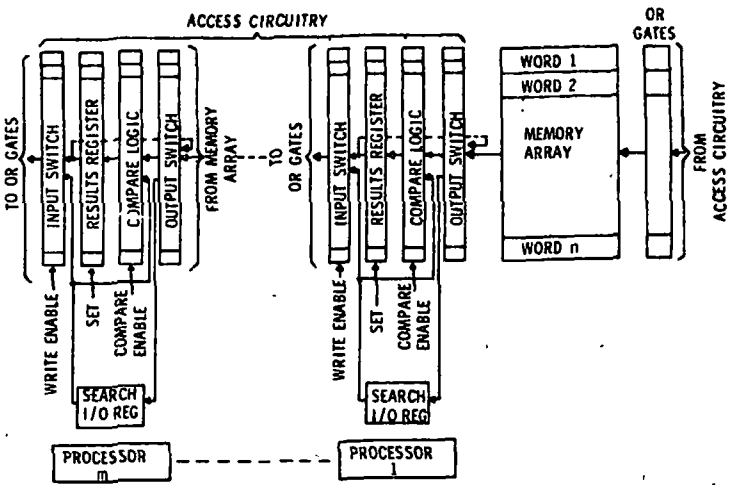


Figure 6. Associative Multi-Access Memory

As was previously suggested, the stored words must be constantly cycled and distributed to the access circuitry of all processors. There are two quite different ways of mechanizing the memory array to meet these requirements. The memory might operate as a shift register with end-around shift capability. However, the memory might also be strobed to obtain readout of each bit slice in turn without being shifted. This latter technique will make hysteresis devices such as magnetic plated wire applicable, while the shifting technique is limited to active devices. The strobing technique has an undesirable reliability feature since an interrogate driver (in the case of plated wire) is required per bit slice. Thus, the failure of one of these drivers would result in the loss of a single bit of all words. The shift pulse for a shift register, on the other hand, would be supplied on a per-word basis so that a failure would affect a single word or group of words depending on how the shift clock pulse is distributed. Of course, the centralized shift pulse generator must be assumed ultra-reliable.

A reliability advantage exists in the associative version of the multi-access memory. This is due to the capability of relabeling words. When a word failure occurs, and assuming that a reloading capability is available, some previously unused word of memory can take the place of the failed word by simply writing the same identity and data that was in the failed word. This prevents the necessity of program changes which would be required if each word had a fixed label or address.

A Distributed Multi-Access Memory -- If the shift register approach to mechanization is used, either the location-addressable or associative versions of the multi-access switch can be organized with the access circuitry distributed throughout the memory array. One complete set of access circuitry can be inserted between each pair of adjacent bit slices of the memory array as shown in Figure 7. The access circuitry (AC) shown in Figure 7 can be of either the associative or location-addressable types.

There are a number of potential advantages of this method of organization of the multi-access memory:

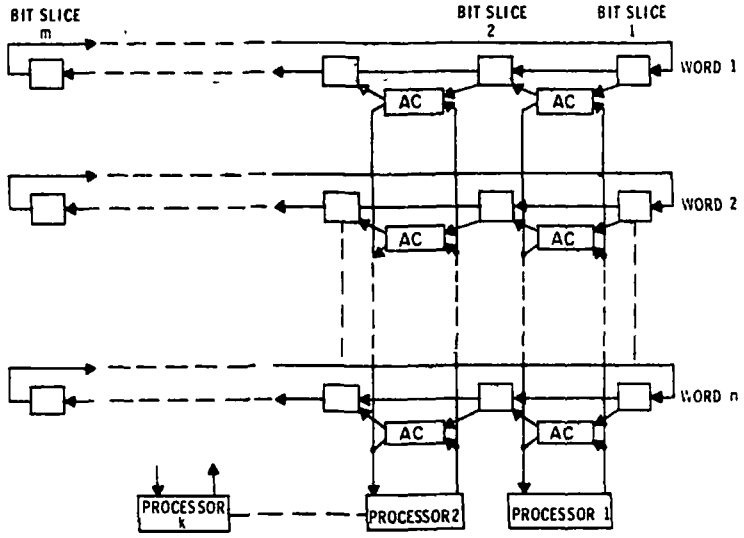


Figure 7. Distributed Multi-Access Memory

- The fanout from each word of the memory to the access circuitry is distributed evenly throughout the word, thus, no special driver is required and each bit of the word is exactly the same.
- The distributed multi-access memory has a structure that would make it suitable for advanced batch fabrication techniques.
- The memory is easily expandable since it is a cellular structure with uniform intercell connections.
- The fact that accessing units are out of step tends to eliminate conflicts, particularly in writing, that could occur if all are in synchronism.

Characteristics of a Multi-Processor Using Multi-Access Memories -- The multi-access memories described above are proposed for use in an ultra-reliable multi-processor system. Probably both location-addressable and associative versions would be used to satisfy all the memory requirements of the system.

Assuming multi-programming and parallel processing at the segment level, an associative multi-access memory would be used to handle functions of the Control Memory previously described. However, it would be more complex than the one previously described since it must be able to perform a maximum search. Operand storage would probably also be in the associative portion. This would eliminate the problem of allocating specific locations for operands and thus make it easier to transfer operands from one program to another.

Program storage requirements would seem to be adequately handled by the location-addressable type of memory since programs, at least as presently organized, are inherently sequential. However, there is a reliability advantage in having associative program storage since loss of a single word will have no effect on the program. Another situation in which an associative memory for program storage would be useful is if parallel processing is

performed at an instruction level rather than at the segment level as has been assumed. In this case a "ready" field, such as that described for state words, is included in each instruction. An instruction is made ready when the operands used by the instruction have been generated, and searches are performed by available processors to find instructions ready to be performed.

The processors of this system would most suitably operate in a serial mode. During each word time of the memory, each processor could obtain one instruction or one operand. This could very reasonably be increased to the extent that an operand and an instruction could be received simultaneously. This would require separate paths from the location-addressable and associative portions of the memory. Given this capability, the next instruction and the operand for the current instruction could be obtained at the same time.

A multi-processor system using the distributed multi-access memory of both the location addressable and associative types and serial processors has the following characteristics:

- Reliability is the main advantage. The distributed multi-access memory allows both the memory and switch functions to be realized with a single cellular array in which single failures will affect only one word of memory or one processor. In the associative portion, the loss of a single word will have little affect, assuming that the information stored in the failed word can be reproduced.
- The system is readily expandable both in terms of the number of processors and in terms of the number of words of memory. The connections from the processor to the array are common to all words, thus the hardware in the processor does not change regardless of the number of words. There is the somewhat unusual restriction that the number of processors in the system cannot exceed the number of bits per word. This does not appear to be a severe restriction.

- Processors operate one or more bit times out of step. This allows parallel processing to a greater degree than would otherwise be possible. A result produced by one processor can be used by another one bit-time later, rather than an entire word time later as would be the case if all processors were in synchronism. This is a significant advantage, however, only if parallel processing is being performed at an instruction level.

In the previous discussion, a distributed location addressable multi-access memory was assumed for program storage. There are some possible advantages in using the undistributed version for this function. For one thing, this would allow the word length to be considerably longer than that required for one instruction. For instance, an entire program segment (100 instructions or so) might be stored in one word of memory. In this case, the access should not be synchronous cyclic as in the previous case, but rather should be asynchronous cyclic, that is, the contents of the word are still required in a specific order but a starting and stopping capability is desired.

The only conclusions that can be drawn regarding the use of multi-access memories in multi-processor systems is that they provide some interesting features, particularly the distributed version of the multi-access associative memory. The feasibility of mechanizing such a memory and its effects on multi-processor system characteristics should be given further attention.

AN ASSOCIATIVE PROCESSOR FOR PICTURE PROCESSING

Spacecraft, and in particular unmanned spacecraft, are often required to transmit pictorial data to data processing centers on the earth. Since each picture contains vast amounts of data, much of which may be redundant in a given application, it is desirable to perform as much picture processing as possible in the spacecraft in order to lower the required bandwidth and thereby lower the power requirements of the spacecraft transmitter.

Picture processing computations are intrinsically parallel. The parallel processing characteristics and high-speed equality search capability of an associative memory make it well adapted to this application. Conventional associative memory instructions can generally be employed, thus requiring very little special external logic. The two-dimensional memory configuration naturally fits the picture format, and processing can proceed in an orderly fashion. Processing speed depends only slightly on the characteristics of the picture.

The use of an associative memory for a particular picture processing problem has been briefly investigated. Given a binary representation of a picture, the problem is to determine the number of irregular shaped objects (clouds) and to determine and record the size (number of 1's) of each.

Basic Method

This method is based on the use of an associative memory containing two sections. One section stores the picture while the other section is used both as a scratchpad and as a data store for the results of the picture processing. The basic operations required are those that can normally be performed by an associative processor such as the equality search, addition, counting, shifting the contents of the Results Store, etc.

The approach to processing is to move across the picture one bit slice at a time from right to left and to use the scratchpad portion of each word to keep track of a cloud identification number and the number of 1's in that row of the current cloud. The end result after one pass over the picture is the number of clouds and the number of 1's in each. The processing operations required are called initiation, continuation, expansion, contraction, and termination. Consider first the operation of continuation. A search for the combination 11 is performed over the current bit slice and the previous bit slice of the picture. All words satisfying the search must have their "number of 1's" field incremented. Similarly, the contraction operation is performed by searching for

01 combinations adjacent to 11 combinations. Whenever a cloud contracts, the cloud identification number and the field containing the number of 1's are removed from the row satisfying the search and the number of 1's must be added to the number of 1's in the adjacent word which has the 11 combination.

The termination operation provides a good example of an operation that makes good use of the associative capabilities. A termination is detected by searching for an 01 combination not adjacent to a 11 combination. Before terminating this cloud and storing its count, a search must be made over all cloud numbers previously stored to see if other portions of this cloud have been previously terminated. Such a situation is possible if, for instance, the cloud is U-shaped.

Continuation, expansion, and contraction are relatively independent of the number and size of clouds in the picture. However, at least a portion of initiation and termination operations must be performed individually for each cloud in the bit slice.

Detail of the Method

An associative processor for this application might be arranged as shown in Figure 8. Only the associative memory portion is shown; in addition to this, a program memory, a control unit, and an input-output section would be required. These latter units would be very similar to their counterparts in a conventional general-purpose computer.

The associative memory is divided into two main sections. The first of these consists of the picture storage area. It is assumed that the picture has been quantized and that a threshold operation has then been performed such that the elements are binary (two-valued). Thus, the presence of a cloud might be represented by a 1, and the absence of a cloud would then correspond to a 0. The search operation is the only operation that is performed on the data stored in this section of the memory. It is performed only on adjacent bit slice pairs, where a bit slice consists of a *single column* of bits.

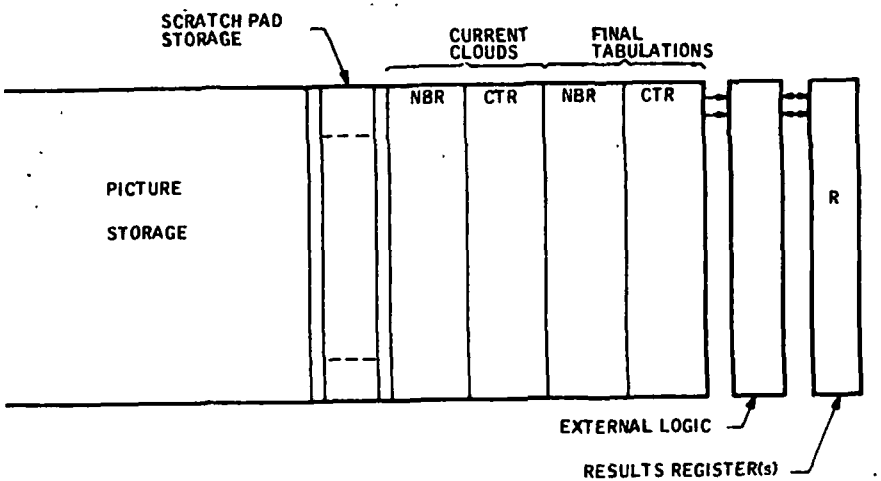


Figure 8. Associative Processor Configuration

The other part of the memory consists of the scratchpad and tabulation portion. The scratchpad area is used to temporarily record the results of processor operations while the tabulation area stores cloud numbers and cloud size information.

The functions performed by the external logic depend on the type of memory element employed. If the element is capable of local logic, fewer functions need be performed by the external logic than if it does not have that capability. The combination of local and external logic must be capable of performing such operations as comparison, logical sum, logical product, and locating the first 1 in the results register.

The results register stores the result of a search or of any of the other processor operations. In actual practice, two registers are required. The second register performs auxiliary functions such as storage of the bit marking the location of the first 1.

Flow charts will be used to explain in some detail the operation of the associative processor in this application. Since no formal list of instructions has been compiled for the processor, the entries in the flow chart boxes consist of somewhat arbitrary symbols and abbreviations selected to represent the operations performed and at the same time to maintain a compact notation. The abbreviations used in the flow charts are as follows:

Symbol	Meaning
R	Results register
A through K	Bit slice scratchpad locations
CTR	Word field in which counter is stored
NBR	Word field in which cloud number is stored
SCH	Search
SH	Shift

Symbol	Meaning
DN	Down
UP	Up
S_2	Bit slice being processed
S_1	Previous bit slice
$X \cap Y$	Logical product of bit slices X and Y
$X \cup Y$	Logical sum of bit slices X and Y
#	A cloud number
$(X) \rightarrow Y$	Replace Y with X or its contents
LOC	Locate

Continuation -- The first three blocks in the flow chart sequence of Figure 9 represent search operations on a pair of adjacent bit slices. The bit slice on the left, S_2, corresponds to the new bit slice while the one on the right, S_1, is the old one from the previous processing operation. If the search speed depends on the number of bits in the field being searched, this process of searching for particular two-bit combinations and storing the results in single-bit fields is advantageous, otherwise the S_2 and S_1 fields could just as well both be searched each time. Of the three useful bit combinations, the pair 01 denotes contraction or termination, the pair 10 denotes initiation or expansion, and the pair 11 denotes continuation. After performing these three searches and storing the results, the latter result is retained in the results register and if any of the bits are 1's, the counters in the corresponding rows are incremented.

Contraction -- The 01 combination stored at location A of Figure 9 is checked next and, if any of these exist, bits corresponding to the locations of the 11 combination, stored at location B, are placed in the results (R) register. The R register is then shifted up and its contents are compared with the set of bits, A, marking the 01 locations.

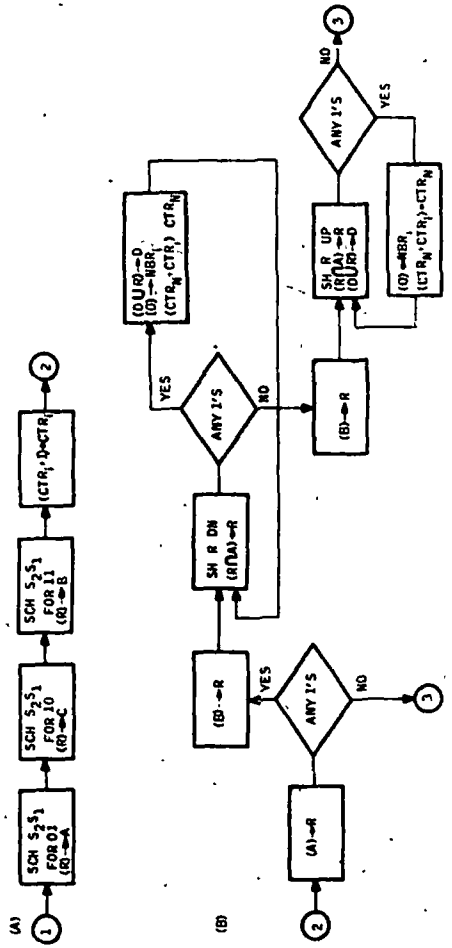


Figure 8. Flow Chart for (A) Continuation and (B) Contraction

12017-FR1

If there is coincidence between any of these sets of bits, it means that the corresponding 01 pair is adjacent to a 11 pair and that the cloud is contracting. The cloud number field in these rows are then cleared, and the contents of their counters are added to the counters in the 11 rows. This cycle is repeated until there are no more coincidences between 01 pairs and shifted 11 pairs. The entire process is then repeated, except that the R register is shifted down. This procedure locates adjacencies in the other direction. Bit slice location D is used to record all of the 01 combinations that are adjacent to 11 combinations. The remaining ones ($A \bigcap \overline{D}$) correspond to cloud terminations.

Expansion -- When all the picture elements corresponding to contraction have been processed, the program enters the expansion phase (Figure 10). The first step consists of a check of the C field for 10 bit pairs. If any of these exist, bits corresponding to 11 locations (stored in the B field) are placed in the R register. The R register is then shifted down and compared with the bits corresponding to the 10 locations (C). If coincidence is detected, the cloud number is inserted in the proper field in the new rows, and the corresponding counters are set to 1. The R register is shifted again, and the process is repeated until coincidence is no longer detected.

During each step of the program for expansion, the R register is shifted up one extra place and compared with the tally of 11 locations (B) in order to determine whether or not the cloud being expanded is connected via the new bit slice to another cloud. If it is, the cloud numbers in the rows associated with the two clouds are made equal, if it is not, the numbers are left unchanged, and the search for expansion bits is continued. Bit slice location E is used to record all of the 10 combinations that are adjacent to 11 combinations. The remaining 10 combinations ($C \bigcap \overline{E}$) represent cloud initiation elements (see Figure 11)

Initiation -- The set $C \bigcap \overline{E}$ is then checked and if any 1's are found, a second operation is performed to find the first 1 in the first group. It and the rest

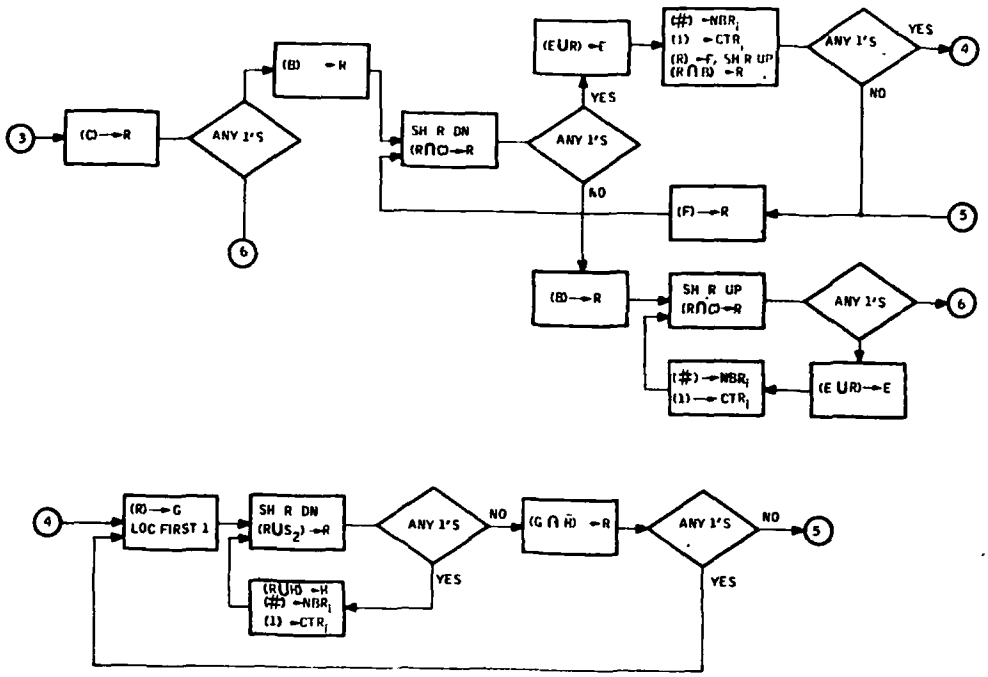


Figure 10. Flow Chart for Expansion

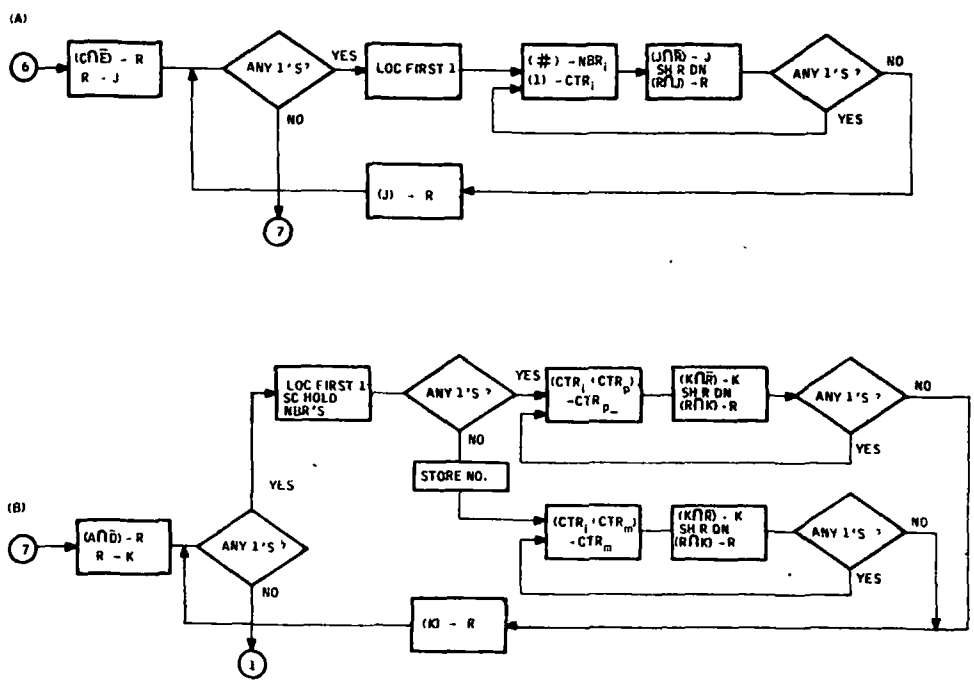


Figure 11. Flow Chart for (A) Initiation and (B) Termination

of the 1's in that group are assigned a cloud number and 1's are placed in the corresponding counter locations. This group of 1's, representing the first new cloud, removed from the set, and the procedure is repeated until all of the elements of $C \bigcap \bar{E}$ have been used

The termination procedure (see Figure 11) begins with a search of the set $A \bigcap \bar{D}$ where A represents all the 01 combinations and $\bar{D}$ represents those that are adjacent to 11 locations. If any 1's are found, a check is made to find the first 1. A search over previously stored cloud numbers is then made to see if it is part of a cloud that was previously terminated. If it is, the previous termination was premature and the contents of the counters in the rows associated with this new part of the cloud must be added to the previously stored count. If it is not part of a cloud that has been previously terminated, the cloud number is stored in a section of the scratchpad memory reserved for output data and the contents of all of the counters in the rows with this particular cloud number are summed and stored in the proper adjacent location. The checking procedure is then continued in the same manner for any additional clouds that terminate in the current bit slice pair. After all terminations are processed one computation cycle is complete, and the program repeats on another bit slice pair--the last received of the current bit slice pair and the adjacent new one

Conclusions

The flow charts of Figures 8 through 11 illustrate one method of using an associative processor to extract certain specified data from a picture. In this case it was desired to count the number of clouds in the picture and to determine and record the size of each cloud. It appears that an associative processor is well adapted to solving this problem and would be able to do the job considerably faster than a general-purpose computer for two reasons:

- A relatively large number of searches are required which must be done sequentially in a conventional memory.
- The other operations (logical and arithmetic) can, in most cases, be done in parallel on all the elements of a single bit slice rather than sequentially as they are done in a general-purpose computer.

Some of the operations have not been completely decomposed into basic instructions and further study would be required to complete this task.

It can be noted that complete flexibility exists since each word of the associative processor is capable of keeping track of its current cloud number and of serving as a counter to record the number of 1's. The use of an associative processor provides other advantages also. For instance, a histogram of cloud sizes can readily be generated when the processing is completed for purposes of generating statistical parameters.

There are various types of memory organizations that could be used in this application. A bit slice write capability, which in turn implies an NDRO memory element, is the main requirement. A two-dimensional readout capability is also required. Local logic is not necessary, and, since most of the searches are over relatively short fields, it would not speed up the operation appreciably.

The picture elements could be stored in a separate memory rather than including them in the processor memory. The requirements for this separate memory would not be as severe as those for the processor memory. A simple sequential access memory with two output registers to hold bit slice pairs would be adequate. It could operate at much lower speeds than those required by the processor memory. Less external electronics is required, however, if the two memories are combined and constructed from the same type of components.

AN ASSOCIATIVE PROCESSOR FOR INCREMENTAL COMPUTATION

A description of methods of using an associative processor for incremental computation is given in Reference 1. Since that report was published, a new improved method of storing the program and of performing the program steps was devised. A description of this method follows a review of the remaining portion of the system.

Incremental computation is usually performed by suitably interconnecting a number of digital integrators. Each integrator computes according to the formula $\Delta Z = k\, Y\, \Delta X$ which when summed forms $\Sigma \Delta Z = Z = k \Sigma Y \Delta X$. This sum is an approximation to the integral $Z = k \int y dx$.

The digital integrator consists of three major parts -- a Y register connected as a counter, an R register and adder-subtracter connected as an accumulator, and the necessary logic for generating the ΔZ and controlling the counter and accumulator from the ΔX and ΔY inputs. Although binary (two-valued) increments are sometimes used, ternary (three-valued) increments are much more common and will be assumed in this application. The three possible incremental values are +1, 0, and -1. Means must be provided to add several ΔY increments into the contents of the Y register. This can be done either sequentially or in parallel. Addition of a single increment to the Y register requires counting up by 1, no change, or counting down by 1. Addition of more than one increment may require several counts. The R register is updated by either adding Y to R, leaving the contents of R unchanged, or subtracting Y from R depending on the value of ΔX. If this is done after the Y register is updated with the most recent ΔY, the new value of Y is used; if it is done before the Y register is updated, the old value of Y is employed. The ΔZ increment consists merely of the overflow or output carry (or borrow) of the R register. It is sensed after each R ± Y operation and then encoded and stored for later use as either a ΔX or ΔY input to other integrators.

ng consists of determining the proper values for the constant k associated each integrator and modifying the integrator such that the output $\Delta Z =$ ΔX. If k can be made equal to a power of two, the scaling process can nplemented by selecting the proper position in the Y register in which to he ΔY increment. In the few cases in which k is not equal to a power o, a separate integrator is used as a constant multiplier. A constant k aced in the Y register, and the ΔY increment is connected to the ΔX in- resulting in an output of $\Delta Z = k\, \Delta Y$.

ramming is accomplished by connecting the ΔZ outputs to the proper ΔX ΔY inputs such that the resulting integrator configuration is constrained lve the appropriate difference equation. In the associative processor anization, this is accomplished by means of searches and addition oper- s.

proposed word format for an associative processor for incremental com- ion is shown in Figure 12. There are a total of nine different fields in ord. Four of these fields are used to store the Y and R registers and X and $\Sigma \Delta Y$ increments. The $\Sigma \Delta Y$ field is a short word that contains um of several increments since it is generally required that an integrator pt several ΔY inputs. A single ternary increment, coded as two binary s, is sufficient to represent ΔX.

four remaining fields are somewhat unconventional and require a more led explanation of their purposes. The scaling control bits are used for ng the various integrators. Each scaling word consists of the binary esentation of the number of positions the $\Sigma \Delta Y$ field must be shifted be- it is added to the Y field. The $\Sigma \Delta Y$ fields are shifted simultaneously in ollowing manner. First, the scaling control bits are searched, one at a , starting with the most significant bit. As each bit location is searched, ΔY field in all the word locations in which a match is obtained are shifted nber of places corresponding to the weight of that bit. After all of the ng control bits have been searched, the $\Sigma \Delta Y$ words are in the proper ions and they are then added simultaneously to the corresponding Y fields.

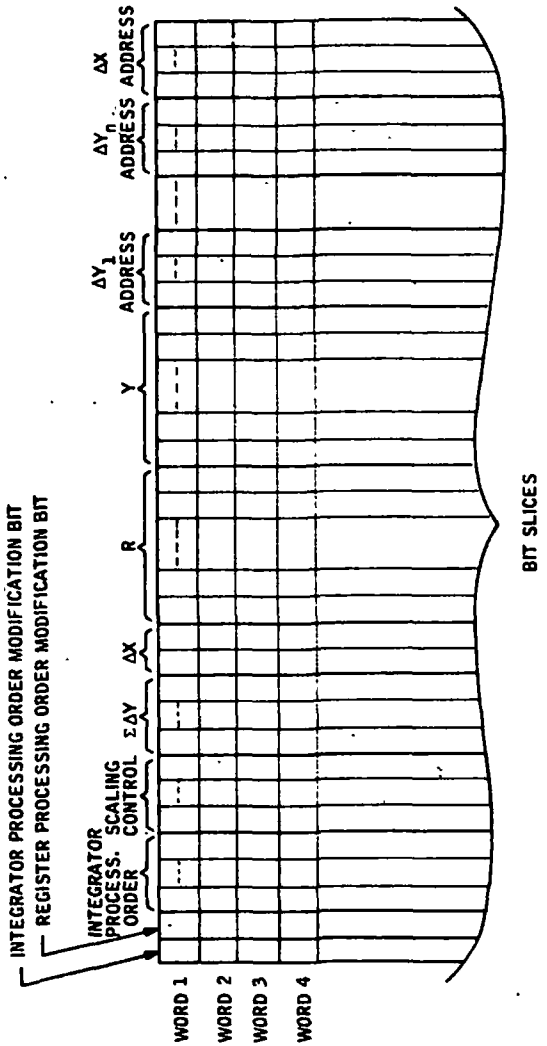


Figure 12. Memory Arrangement for an Associative Memory Used for Incremental Computation

ome algorithms, certain sets of integrators must be processed sequentially ause of the nature of the difference equation being solved. The order of cessing depends on whether the particular part of the equation being con- ered calls for the use of the old value of Y or the new value of Y. The old ue of Y corresponds to the contents of a Y register that has not been up- d during the present iteration period, while the new value of Y corresponds he contents of a Y register that has been updated. The natural method of ration of this type of associative memory results in the consistent use of old value of Y since the integrators are processed in parallel. Therefore, rder to use the new value of Y, it is necessary to set aside one of the nory word fields to establish the order of integrators that must be pro- sed sequentially. These bits are so indicated in Figure 12.

sometimes necessary to modify the processing order of sequentially cessed integrators, depending on the sign of the ΔX increment or other erion. This modification is obtained by means of search operations per- ned on the processing order bits and on the ΔX bits. A single control bit been added for this purpose. It permits a specified criterion to control processing order. If more than one criterion is to be used, additional rol bits must be added. Some algorithms also require that the order of cessing the Y and R registers be interchanged. An additional control bit also been added for this purpose.

programming fields on the right-hand side of the word contain the ad- sses of the ΔY and ΔX fields into which ΔZ outputs originating in a par- lar integrator are to be added. The ΔY address fields are simultaneously rched as the addresses in the corresponding fields of the search register incremented. The ΔZ bits stored in the two most significant bit locations ue R register are added to the Σ ΔY fields in each word in which a match dicated. The same process is then repeated while the ΔX address field earched, and the ΔZ bits are then stored in the appropriate ΔX fields. ΔZ bits are then cleared to zero.

SECTION III

EVALUATION OF ASSOCIATIVE MEMORIES FOR SPACE APPLICATIONS

Associative memories have been found to be applicable to a number of problems on a space vehicle. These applications are summarized here in terms of the appropriate associative memory organizational approach, the associative and processing capabilities required, and such items as speed, size, etc. From this summary, a number of organizational approaches are selected as being of interest for space exploration vehicles. The device recommendations discussed in the next section are made for each of the selected organizational approaches.

Since certain of the associative memory organizational approaches defined in Reference 1 are referred to in this section, the definitions of those of interest are repeated below:

2a - This is an associative memory with no local (distributed) logic, with random access to a bit slice for search or processing operations, and with a full adder per word to increase speed of arithmetic operations.

2c - This approach also has no local logic but is significantly different than 2a because it has cyclic access to bit slices. This approach is not suitable for processing operations and thus has a minimum logic capability external to the array for the equality search operation.

3 - This approach has distributed logic to the extent that each cell can compare its own contents to a corresponding bit of the search register. With proper output signals from the cell, an all-parallel equality search can be performed. It also does reasonably well at processing operations and therefore is a good general-purpose associative memory.

3a - This approach is identical to 3 except that it does not have a bit slice writing capability.

4 - This organizational approach is much like 3 in that it has local logic and is capable of doing an all-parallel equality search. However, it is able to perform the inequality search and processing operations slightly faster than approach 3 since the cell is somewhat variable in function and the output is a ternary signal.

5 - This approach is the most complex of any defined in terms of the capabilities of the cell. The most significant feature of the cells, each of which contains a full adder stage, is that they can intercommunicate.

JMMARY OF APPLICATIONS

The applications discussed below have been investigated to various degrees - some have been considered extensively and some only slightly.

Incremental Computation

An associative memory for incremental computations, such as occur in navigation and control problems, was described in Reference 1. Some changes in the technique for using the associative memory for this function are included in Section II.

The requirement was found to consist primarily of processing operations with an occasional equality search. Organizational approach 2a is considered the best choice for this application.

Picture Processing

Picture processing was identified as an application on which the parallel processing capability of an associative processor can be applied effectively. A complete description of this application is given in Section II.

A portion of a particular picture processing problem was chosen to determine the requirements on the associative processor. The problem was to determine the number of clouds and the size of each, given the binary representation of a cloud picture.

The method chosen for organizing an associative processor for this problem is to divide it into two parts - one part for storage of the picture and one for processing and storage of results. The part used for storage of the picture needs only an equality search capability. Also, since the searches are always performed over two bits and in a cyclic manner from one edge of the picture to the opposite edge, a form of organizational approach 2c is suitable for this requirement. Accesses to bit slices can be cyclic, but they cannot be synchronous. This is because the time required for processing each bit slice varies depending on the data. Thus an asynchronous cyclic access such as could be obtained from shift registers is required. The number of words required for storage of the picture will not exceed 800.

The part used for processing and storage of results must have quite different capabilities. It must be able to do processing as well as search operations. It also must have facilities for shifting the contents of the results register up or down, and it must have a word-select ladder which allows selection of multiple matches, one at a time. The organizational approach chosen for this part of the picture processing application is 2a. The number of words required is the same as for storage of the picture.

ptive Data Acquisition

application, formerly called adaptive sampling, is that of using an asso-
ve processor as an on-line data collector and pre-processor for sensor
system-status-monitoring data. It is described in detail in Section II.

fly, the Associative Data Acquisition (ADA) system works as follows.
a data source is connected permanently to one word of the associative
essor. During each cycle, the associative processor computes a figure
erit simultaneously for each source and performs search operations to
rmine which source has the most significant data. The output of the ADA
em is this selected value. Thus, it can be seen that the data acquisition
be made to match the communications channel.

capabilities required to accomplish this function are primarily of the
essing type. However, the need for occasional equality, inequality, and
imum (or minimum) searches is also anticipated. Also since the capability
ndling analog sensors is desired, a built-in analog-to-digital conversion
equired. Other possible useful features include the logic to determine the
t number of matches (number of 1's in the results register) and a word-
ct ladder to select multiple matches one at a time for readout.

organizational approach most suitable for this application is 2a. In most
s the number of words required would be less than 1000.

rol Functions in a Multi-Processor System

application is also discussed in detail in Section II. In a multi-processor
em which has multi-program and parallel processing capabilities, an
ciative memory is proposed to handle those functions such as assignments
sks to processors, assignment of memory to programs, etc., which are
ially handled by an executive control program.

An associative memory with search capabilities only can satisfy the requirements of this application. An all-parallel equality search capability of approach 3 can probably be justified, and the number of words required would be less than 1000.

Look-Aside Associative Memory

This application is described in Section II. An associative memory is attached to each processor in a multi-processor system and is used to store instructions and operands obtained from main memory in anticipation that they may be needed again soon. Results are also stored in the associative memory rather than in the main memory in anticipation that they may be required as operands for future instructions. Each time an instruction or operand is needed, a search is performed to see if it is in the associative memory before accessing main memory. Thus, the purpose of the look-aside associative memory is to save accesses to main memory.

The requirements of this application include an all-parallel equality search, very high speed, and fairly small size. No processing operations are required. There is also no need for a bit-slice writing capability or for a word-select ladder. Thus, organizational approach 3a is recommended.

Multi-Access Associative Memory

In Section II, a multi-access associative memory is proposed for use in a multi-processor system. This memory can simultaneously satisfy the requests of many users, although each obtains its requested word in a serial-by-bit mode. This is accomplished by distributing the contents of the memory serially to the many sets of access circuitry in a cyclic synchronous mode.

Although the multi-access memory can be used to perform a number of different functions in a multi-processor system, none require processing operations and most require only the equality search. Also, the serial-by-bit mode of operation is suitable not only for the searching operation but also for reading and writing. The number of words required depends on the specific application. If it is functioning only as a control memory, it would be small, if it is used for program and operand storage, it could be quite large. In either case organizational approach 2c would be used.

Sum of Products

In Task 1 of this study, the computational requirements of space exploration were described in terms of a set of computation descriptors. The computation descriptor found to occur most frequently was the sum of products. For example, the sum of products operation showed-up frequently in data compression techniques, it was the major part of the computation required to generate a picture from side-looking radar inputs, it occurs in many picture processing or pattern recognition algorithms and even occurs occasionally in navigation and control type computations. Although this operation has not been given a great deal of attention in this study, a few comments about the problems encountered in performing it in an associative processor are in order.

First, there are two variations of the sum of products operation to be considered. In one case, the summation is over a regular subset of words in the associative processor. A summation over all words or over every other word are examples of regular subsets. In this case, an associative processor can handle the job very well. First the multiplication is performed over the selected subset of words, and then the summation is performed by a sequence of adds where pairs of words in the selected subset are summed (all pairs simultaneously), then pairs of these sums are summed, etc. Thus, for n words, a total of $\log_2 n$ adds are required. This is a fairly efficient way of performing this variation of the sum of products operation.

The second variation of the sum of products operation is one where an irregular subset of words is involved in the summation. While the multiplication part of the computation can be done simultaneously over all words of the selected subset, the summation presents a problem. Since the words to be summed are stored in random locations, it is not possible to form sums of pairs simultaneously for all pairs as was done in the previous case. One technique proposed for handling this type of summation uses an external accumulator and requires the capability of determining the exact number of 1's in the results register (Ref. 7). The procedure is to search for 1's in the least significant bit position of all words. The number of 1's is then added into the external parallel accumulator. The next step is to search for 1's in the second bit position of all words, again determine the number of 1's in the results register, and add the number into the accumulator shifted one position to the left with respect to the previous step. This procedure is continued through the most significant bit, at which time the summation of all the numbers is contained in the accumulator.

It can be seen that the effectiveness of this technique depends on a fast way of determining the number of 1's in the results register. The associative processor suitable for this application is one strong in processing operations. Organizational approach 2a is probably the best choice. It can be expected that most requirements would be satisfied with an associative processor of less than 1000 words.

Retrieval from a Large File

Since an associative memory's most significant capability is that of performing a parallel-by-word equality search, a purely information retrieval application is postulated. Such a requirement might simply be that of selecting a subset of quantities from a mass storage device for transmission to earth. Assuming that the quantities are stored at random locations on the mass storage device, a convenient way of obtaining all the quantities in one pass is to

ore the identity of all the desired quantities in an associative memory. As ch quantity is received from the mass storage unit, it serves as the search ord in an equality search operation to determine if it is one of the desired antities. This requirement can be met with a very simple associative emory. The only operation needed is the equality search. Also, multiple atches will never occur, and input/output consists only of initial loading. ssuming that the quantities are received from the mass storage unit in a rallel-by-bit mode, an all-parallel equality search is desirable. Thus, ganizational approach 3a is recommended. It is likely that 100 words uld be a sufficient size for such applications.

atrix Inversion

ie requirement for matrix inversion was also found to occur occasionally space exploration. This problem is of the type that can take advantage of e parallel processing capability of an associative processor. The Gauss-rdan method, which consists of simply diagonalizing the matrix by the ocess of elimination and generating the inverse by performing the same erations on an initially diagonalized matrix, was the algorithm chosen. ie time required to perform this operation on an associative processor is oportional to n^2, where n is the size of the matrix, while in a sequential mputer the time is proportional to n^3.

ie capabilities required to perform this computation are predominantly of e processing type. Floating point capabilities would be useful, which, cording to the algorithm described in Reference 1, necessitate extensive ifting operations. An occasional equality and inequality search are used select the proper subset of word for the processing operations. Due to e shifting requirement, organizational approach 5 is the best choice. If e floating point capability was not required, approach 2a would be chosen. ie number of words required for this application is $2n^2$.

Applications that use similar procedures are the simplex method of linear programming and integer linear programming problems. Although requirements for these computations were not presently identified in space exploration, they could occur in the future.

Advanced Associative Processor

An associative processor with powerful processing capabilities as well as all-parallel search capabilities would, of course, satisfy all the applications described above. This general-purpose associative processor would best be realized by approach 5, which has a complex cell and intercommunications between cells. Organizational approach 4, with its ternary output from the cell and a limited variability in the function performed by the cell is another possibility. It does not have the processing capabilities of approach 5, but it has the advantage that it can be mechanized with magnetic devices.

EVALUATION OF APPLICATIONS

The applications for associative memories are summarized in Table 1. The only undefined notation in the table is concerned with size or capacity. A memory of less than 1,000 words is considered small and designated with S, from 1,000 to 10,000 words is considered medium and designated with M; large memories are designated with an L. The numbers in the "Preferred Organizational Approach" column refer to definitions at the start of this section. The following conclusions are drawn from the information in Table 1:

- Two organizational approaches stand out as being the most useful for space problems. One of these is a bit-slice processing approach with a serial adder per word (approach 2a) which is useful where extensive processing is involved. The other is capable of an all-parallel equality search (approaches 3 or 3a) and is useful for applications requiring mostly searching.

- In nearly every case the associative memory required is one of less than 1,000 words.
- While the requirements for none of the applications can be completely satisfied with a read-only associative memory, there are some situations where a read-only field could be utilized
- Device recommendation will be made for four organizational approaches - 2a, 2c, 3, and 5. Approach 3a is not included individually since any device recommendations for approach 3 will hold for 3a also.

Table 1. Applications Summary

APPLICATIONS OR FUNCTIONS / REQUIREMENTS		SEARCH OPERATIONS: EQUALITY SEARCH	SEARCH OPERATIONS: INEQUALITY SEARCH	SEARCH OPERATIONS: MAX OR MIN SEARCH	SEARCH OPERATIONS: UNION OF SEARCHES	SEARCH OPERATIONS: INTERSECTION OF SEARCHES	PROCESSING OPERATIONS: ADD (X_1, S, X_2, S, ... X_n, S)	PROCESSING OPERATIONS: ADD (X_1, Y_1, X_2, Y_2, ... X_n, Y_n)	PROCESSING OPERATIONS: COUNT	PROCESSING OPERATIONS: SHIFT	COMPLEX OPERATIONS: PROXIMITY SEARCH	COMPLEX OPERATIONS: FLOATING POINT OPERATIONS	COMPLEX OPERATIONS: SUMMATION (X_1, X_2, ... X_n)	FACILITIES FOR PROCESSING OF RESULTS: ONE DIMENSIONAL SHIFT CAPABILITY	FACILITIES FOR PROCESSING OF RESULTS: TWO DIMENSIONAL SHIFT CAPABILITY	FACILITIES FOR PROCESSING OF RESULTS: LOGIC TO DETERMINE EXACT NO. OF MATCHES	FACILITIES FOR PROCESSING OF RESULTS: WORD SELECT LADDER	OTHER REQ: EXTREME SPEED REQMT.	OTHER REQ: CAPACITY (NO. OF WORDS)	OTHER REQ: READ ONLY FIELD POSSIBLE	MOST FREQUENT TYPE OF OPERATION	PREF ORG APP: BEST	PREF ORG APP: NEXT BEST
MENTAL COMPUTATION		*					*	*	*													PROCESSING	2a	
RE	(PICTURE STORAGE)	*																		S/M		SEARCH	2c	
SSING	(SCRATCH PAD)	*				*	*		*			*		*				*		S/M		PROCESSING	2a	3
TIVE DATA ACQUISITION		*	*	*			*	*					*				*	*		S/M	*	PROCESSING	2a	
OL FUNCTIONS IN A -PROCESSOR SYSTEM		*	*	*		*												*		S	*	SEARCH	3	
ASIDE ASSOCIATIVE RY		*																		*	S	SEARCH	3a	
-ACCESS ASSOCIATIVE RY	FOR CONTROL FUNCT'NS	*	*	*														*		S		SEARCH	2c	
	FOR PROGRAM AND OPERAND STORAGE	*																		M/L		SEARCH	2c	
F PRODUCTS COMPUTATION		*						*				*		*		*				S/M			2a	
VAL FROM A LARGE FILE		*																		*	S	SEARCH	3a	
X INVERSION		*	*				*	*	*			*		*						M		PROCESSING	5	2a
CED GENERAL PURPOSE ATIVE PROCESSOR		*	*	*	*	*	*	*	*	*		*		*	*	*	*			S/M		EITHER	5	3

SECTION IV
DEVICE EVALUATION

Many devices can be used to implement an associative memory cell. In some organizational approaches, the cell requires both memory and logic functions, while for others only the memory function is required. Some of the devices potentially usable in an associative memory cell have become obsolete in recent years due to advances in electronics and the discovery of new and better elements; others have been improved to the point where they are currently being used in computer memories, still others are potentially very desirable elements but cannot be used at this stage in their development due to unsolved fabrication problems.

The purpose of this portion of the report is to select from among this group of devices, a subset whose members are either usable in associative memories at the present time or are potentially usable in the future. This is a difficult task for two reasons. First, many parameters can influence the desirability of a particular component, second, each of the various associative memory organizations places differing requirements on the components. Because of this, a component rating system was devised to facilitate the evaluation of the components with regard to the requirements of the various memory organizations.

RATING SYSTEM

The heart of the component rating system is a set of 14 component parameters, each of which is related in some way to an important memory or logic characteristic. In most cases the components can be given a merit value for each of these parameters which is independent of the size and type of memory organization being considered. Similarly, a set of weights, one for each of the

component parameters, is assigned to each memory organization. The magnitude of each weight depends on the importance of the corresponding parameter to the over-all functioning of an associative memory with this particular organization. The sum of the products of the component merit values and the associated weights for a particular memory organization is interpreted as a figure of merit for this component when it is used in the particular organization. The figures of merit for all of the components relative to this particular organization can then be compared in order to select those components with the highest figures of merit.

Some of the parameters, however, do not appear to properly fit into this type of computation. These are the parameters that correspond to various confidence levels, and it appears that they should be multiplied by the figure of merit rather than being included in the sum of products computation. The final computation then is of the form

$$F = \prod_j c_j \left(\sum_i w_i v_i \right)$$

where the w_i are weights corresponding to a particular organization, the v_i are parameter merit values for a given component, the c_j correspond to confidence level figures, and F is the resulting figure of merit for the component used in the particular memory organization.

The values of w_i and v_i range from 0 to 10 while the magnitude of the c_i range from 0 to 1. As a result, the maximum figure of merit for an element is 100n, where n is the number of relevant parameters.

COMPONENT PARAMETERS

The component parameters that have been selected are given in Table 2, and a brief discussion of these parameters follows. The read power is a component parameter which in part determines the total power required

Table 2. Component Merit Values

COMPONENT MERIT VALUES PARAMETER	PLATED WIRE (WSA) (1)	PLATED WIRE (WOSA) (2)	PLANAR THIN FILM	BIPOLAR INTEGRATED CIRCUITS	FIELD EFFECT INTEGRATED CIRCUITS	TUNNEL DIODES	THIN FILM TRANSISTOR CIRCUITS	CRYOTRONS (3)	FERROELECTRICS	FERROTRON	DIODE-CAPACITOR	FERRITE SHEET	BIAX
A READ (OR SHIFT) POWER	6	6	6	4	6	3	6	2	6	4	3	6	5
B WRITE POWER	3	3	3	2	2	1	2	2	6	6	2	3	2
C READ SPEED	7	7	7	8	7	10	7	6	3	1	6	6	7
D WRITE SPEED	6	6	6	8	6	10	6	6	3	1	6	6	3
E VOLATILITY	10	10	10	0	0	0	0	10	10	10	0	10	10
F SIZE	7	7	8	4	6	4	6	2	10	9	8	8	7
G CAPABILITY OF WORD SLICE READOUT	7	6	8	10	10	10	10	10	6	6	3	6	8
H RELIABILITY	7	9	7	4	5	4	6	7	5	5	4	7	7
J DATA REGENERATION REQUIREMENT	10	10	10	10	10	10	10	10	10	10	0	10	10
K COST (PER CELL)	7	9	7	1	3	1	3	5	8	7	5	8	3
L OUTPUT SIGNAL LEVEL	3	10	1	10	10	10	10	10	7	7	5	4	4
M NDRO CAPABILITY	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Q	N	N
N PRODUCTION CONFIDENCE LEVEL	1.0	1 2	1.0	1.0	1.0	1.0	0 5	0.2	0.5	0.5	1.0	0 9	1 0
P LOGIC CAPABILITY CONFIDENCE LEVEL	0.3	0.3	0 1	1.0	1 0	0	1 0	1.0	0 3	0 5	0	0	0

KEY N-NO (1) WITH SENSE AMPLIFIER
Y-YES (2) WITHOUT SENSE AMPLIFIER-CLOSED SENSE/WRITE LOOP
Q-QUALIFIED (3) ASSUMING SMALL/MEDIUM SIZE MEMORY WITH REFRIGERATOR

ring a memory search operation. Similarly, the write power is a factor it partially determines drive power required to write a bit slice. These wer figures also include any steady state power consumed by the memory mponents. The next parameter, the element read speed relates to the mory search speed while the write speed corresponds to the speed at ich a bit slice can be written.

latility refers to the retention of information during a power failure. A n-volatile storage element (i.e., one that retains information during a wer failure or interruption) is given a large merit value for this parameter, ile a volatile element is assigned a much smaller one. The size parameter rtains to the size of the memory element as well as any associated logic it may be required by the memory cell.

adout of a selected word can always be done serially by means of external ing logic. In most cases, however, it is desirable to read out the bits of a ected word in parallel. The ability or relative ease of obtaining this aracteristic gives a measure of the parallel word readout capability rameter.

e reliability of an associative memory depends on several factors. First, re is the intrinsic reliability or failure rate of the memory element elf. Another reliability factor is the result of the requirement for inter- ment connections. If these connections can be made by means of threading ntinuous wires such as in the case of memory cores, the connections are remely reliable. However, if separate soldered or welded connections ist be made, as in the case of either integrated or discrete semiconductor ments, the over-all memory reliability is reduced due to the failure rate these interelement connections. The amount and type of external drive d sense circuitry required by the element is also a factor in determining over-all memory reliability. All of these factors are considered in signing a merit value to the reliability parameter.

Certain types of memory elements will retain the stored information only for a finite length of time, after which it must be read out and rewritten into the element. This must be done periodically, and during this operation the memory cannot be used for searching and other associative memory functions. Thus, the over-all speed of the memory is compromised, and the restriction could be quite severe in high-speed, real-time applications. This characteristic of a memory element is measured by the regeneration requirement parameter. A merit value of 10 implies that regeneration is not required, while a low value indicates that the memory information must be regenerated frequently.

The cost per bit of the memory is a factor that partially depends on the memory size since it includes not only the cost of the memory element but also the prorated cost of the input and output electronics. Thus, the values assigned to this parameter should correspond to the cost for a typical memory of a given type. The output signal level parameter is important for several reasons. It determines the complexity, size and power consumption of the sense electronics. In memories in which local logic is performed, it can also have an effect on the reliability of the logic operations and the ease with which they can be performed. A merit value of 10 on this parameter implies that the amplitude of the output of a cell is sufficient to drive another cell within the same word line.

The capability of the memory to perform non-destructive read out (NDRO) operations is required for certain of the organizations. In these cases, the parameter relating to this capability is then omitted from the tabulation since all of the elements considered in these cases must have this capability and it is either a full valued or a non-existant quantity. In other memory organizations, either NDRO or DRO (destructive read out) operation is feasible. In these cases, the capability of NDRO operation increases the speed of searches and other operations. Thus, for those elements with the NDRO capability, a larger weight on the speed parameter can be used, whereas a lower weight must be used for those elements that can be operated only in the DRO mode.

The last two parameters represent confidence factors. The first of these is a parameter that expresses the relative confidence that the device will be realized to the extent that it is usable as a memory element in an associative memory cell. The second one is an expression of the confidence that it can be used to perform the logic function required of the associative memory cell. If these capabilities are presently available, a confidence level of 1 is assumed. If there is no possibility of the capability being realized, a confidence level of 0 is used. Figures in between 0 and 1 are used to express varying degrees of confidence in the capability.

ICE RATINGS

The devices have been divided into two groups, one for those that are usable only in cyclic access memories and the other for all of the remaining devices. The first group is rated for organizational approach 2c while the second group of devices is rated for approaches 2a, 3, and 5. These are the four organizational approaches that appear to be the most useful in the applications considered in this study. All the devices considered are described in detail in Reference 1.

The component merit values for the two groups are given in Tables 2 and 3. Although most of the entries in these tables are self explanatory, a few of them require a brief comment.

Two types of plated wire memories are listed. The first is the conventional type with sense amplifiers and drivers on each word line. The second type employs a closed circuit word line (plated wire) which if successful, would permit direct storage of the results of a search in another cell (on the same wire) without the need for sense amplifiers or word drivers. Although this method of operation would provide many advantages in terms of lower cost:

This work is sponsored by the U.S. Army Electronics Command, Fort Monmouth, New Jersey under contract DA 28-043 AMC - 01305 (E)

Table 3. Component Merit Values for Devices Suitable for Approach 2c

Parameter \ Component	Bipolar I. C. Shift Register	Field Effect I. C. Shift Register	Magneto-Strictive Delay Line	Glass Delay Line	Stain Wave (Electro-)	Stain Wave (Magnetic)	Thin Film (Magnetic) Shift Register
A. Read (or Shift) Power	2	4	10	10	8	7	4
B. Write Power	9	10	6	6	7	6	4
C. Read/Write Speed	8	6	7	10	8	8	6
E. Volatility	0	0	0	0	10	10	10
F. Size	8	10	3	5	10	10	8
H. Reliability	5	7	10	10	9	9	9
J. Data Regeneration Requirement	10	10	0	0	10	10	10
K. Cost (per Cell)	6	10	2	10	8	8	4
L. Output Signal Level	10	10	3	3	5	2	1
N. Production Confidence Level	1.0	1.0	1.0	1.0	0.5	0.7	1.0

wer power, and higher reliability, particularly in large memories, the chniques have not been proven and it is therefore given a low confidence vel (Table 2).

ıe figures given for field-effect and thin-film transistor circuits have been ·vised to include recently reported work on complementary symmetry rcuits (Ref. 8). The use of this type of circuitry results in a much higher ›eed/power ratio than could otherwise be obtained. Reported speeds ›tained by the use of this technique are within a factor of 1/2 to 1/3 of those ›tainable from bipolar integrated circuits.

ıe figures for the cryotron include those for the refrigerator. For the emory size assumed (1000 words, 50 bits), most of the cost, size, and ›wer consumption are due to the refrigerator. Thus the resulting figures › not compare as favorably with those for the other components as they ›uld for a larger memory size.

ıe figures for the ferroelectrics and the ferrotron are based on the use of ·w materials which have memory parameters that are much improved over ose previously available. The capability of batch fabricating large memories om these new materials has not been proven, however, which resulted in ·latively low confidence levels.

ŒS OF MERIT

he figures of merit for the four organizational approaches that were selected ·e given in Tables 4 through 7. The weights, individual products, and sums products along with the final figures of merit are shown in Tables 8 rough 11.

Table 4. Figures of Merit for Approach 2a.

Component	$\Sigma w_i v_i$	Figure of Merit
Plated Wire (WSA)	370	370
Plated Wire (WOSA)	404	81
Planar Thin Film	371	371
Bipolar Integrated Circuits	311	311
Field Effect Integ. Circuits	314	314
Tunnel Diodes	296	296
Thin Film Transistor Circuits	322	161
Cryotrons	335	67
Ferroelectrics	324	272
Ferrotron	262	136
Diode-Capacitor	196	196
Ferrite Sheet	345	314
Biax	325	325

Table 5. Figures of Merit For Approach 3.

Component	$\Sigma w_i v_i$	Figure of Merit
Plated Wire (WSA)	329	99
Plated Wire (WOSA)	388	39
Planar Thin Film	326	33
Biplar Integrated Circuits	304	304
Field Effect Integ. Circuits	320	320
Thin Film Transistor Circuits	330	165
Cryotrons	354	71
Ferroelectrics	310	46
Ferrotron	282	42

Table 6. Figures of Merit For Approach 5

Component	$W_i V_i$	Figure of Merit
Bipolar Integrated Circuits	393	393
Field Effect Integ. Circuits	405	405
Thin Film Transistor Circuits	413	207
Cyrotrons	409	82

Table 7. Figures of Merit For Approach 2c

Component	$\Sigma W_i V_i$	Figure of Merit
Biplar Integ. Crt. Shift Register	319	319
Field-Effect Integ. Crt. Shift Register	400	400
Magnetostrictive Delay Line	312	312
Glass Delay Line	386	386
Strain Wave (Electro Acoustic)	442	221
Strain Wave (Magneto Acoustic)	418	293
Thin Film Magnetic Shift Register	341	341

Table 8. Component Merit Values for Approach 2a

Parameter \ Component	Weight	Plated Wire (WAS) (1)	Plated Wire (WOSA) (2)	Planar Thin Film	Bipolar Integrated Circuits	Field Effect Integrated Circuits	Tunnel Diodes	Thin Film Transistor Circuits	Cryotrons (3)	Ferroelectrics	Ferrotron	Diode-Capacitor	Ferrite Sheet	Biax
A. Read (or Shift) Power	5	30	30	30	20	30	15	30	10	30	20	15	30	25
B. Write Power	5	15	15	15	10	10	5	10	10	30	30	10	15	10
C. Read Speed	9/7	63	63	63	72	63	70	63	54	27	9	42	42	63
D. Write Speed	9/7	54	54	54	72	54	70	54	54	27	9	42	42	27
E. Volatility	4	40	40	40	0	0	0	0	40	40	40	0	40	40
F. Size	3	21	21	24	12	18	12	18	6	30	27	24	24	21
G. Capability of Word Slice Readout	2	14	12	16	20	20	20	20	20	12	12	6	12	16
H. Reliability	8	56	72	56	32	40	32	48	56	40	40	32	56	56
J. Data Regeneration Requirement	5	50	50	50	50	50	50	50	50	50	50	0	50	50
K. Cost (per Cell)	3	21	27	21	3	9	3	9	15	24	21	15	24	9
L. Output Signal Level	2	6	20	2	20	20	20	20	20	14	14	10	8	8
M. Ndro Capability	Y/N	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Q	N	Y
N. Production Confidence Level		1.0	0.2	1.0	1.0	1.0	1.0	0.5	0.2	0.5	0.5	1.0	0.9	1.0
Total without confidence level factors included		370	404	371	311	314	296	322	335	324	272	196	345	325
Final figure of merit		370	81	371	311	314	296	161	67	162	136	196	314	325

Table 9. Component Merit Values for Approach 3

Parameter \ Component	Weight	Plated Wire (WSA) (1)	Plated Wire (WOSA) (2)	Planar Thin Film	Bipolar Integrated Circuits	Field Effect Integrated Circuits	Thin Film Transistor Circuits	Cryotrons (3)	Ferroelectrics	Ferrotron
A Read (or Shift) Power	4	24	24	24	16	24	24	8	24	16
B Write Power	2	6	6	6	4	4	4	4	12	12
C Read Speed	6	42	42	42	48	42	42	36	18	26
D Write Speed	2	12	12	12	16	12	12	12	6	2
E Volatility	4	40	40	40	0	0	0	40	40	40
F Size	2	14	14	16	8	12	12	4	20	18
G Capability of Word Slice Readout	7	49	42	56	70	70	70	70	42	42
H Reliability	10	70	90	70	40	30	60	70	50	50
J Data Regeneration Requirement	4	40	40	40	40	40	40	40	40	40
K Cost (per Cell)	2	14	18	14	2	6	5	10	16	14
L Output Signal Level	6	18	60	6	60	60	60	60	42	42
M NDRO Capability	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
N Production Confidence Level		1.0	0.2	1.0	1.0	1.0	0.7	0.2	0.5	0.5
P Logic Capability Confidence Level		0.3	0.3	0.1	1.0	1.0	1.0	1.0	0.3	0.5
Total Without Confidence Level Factors Included		329	388	326	304	320	330	354	310	282
Final Figure of Merit		*99*	*39*	*33*	*304*	*320*	*165*	*71*	*46*	*42*

Table 10. Component Merit Values for Approach 5

Parameter \ Component	Weight	Bipolar Integrated Circuits	Field Effect Integrated Circuits	Thin Film Transistor Circuits	Cryotrons (3)
A. Read (or Shift) Power	5	20	30	30	10
B. Write Power	5	10	10	10	10
C. Read Speed	6	48	42	42	36
D. Write Speed	6	48	36	36	36
E. Volatility	2	0	0	0	20
F. Size	3	12	18	18	6
G. Capability of Word Slice Readout	7	70	70	70	70
H. Reliability	8	32	40	48	56
J. Data Regeneration Requirement	8	80	80	80	80
K. Cost (per Cell)	3	3	9	9	15
L. Output Signal Level	7	70	70	70	70
M. NDRO Capability	Y	Y	Y	Y	Y
N. Production Confidence Level		1.0	1.0	0.7	0.2
P. Logic Capability Confidence Level		1.0	1.0	1.0	1.0
Total without confidence level factors included		393	405	413	409
Final figure of merit		393	405	207	82

Parameter \ Component	Weight	Bipolar I.C. Shift Register	Field Effect I.C. Shift Register	Magneto-Strictive Delay Line	Glass Delay Line	Strain Wave (Electro-)	Strain Wave (Magneto-)	Thin Film (Magnetic) Shift Register
A Read (Or Shift) Power	8	16	32	80	80	64	56	32
B Write Power	7	63	70	42	42	49	42	28
C Read/Write Speed	8	64	48	56	80	64	64	48
E Volatility	3	0	0	0	0	30	30	30
F Size	5	40	50	15	25	50	50	40
H Reliability	10	50	70	100	100	90	80	90
J Data Regeneration Requirement	5	50	50	0	0	50	50	50
K Cost (per Cell)	5	30	50	10	50	40	40	20
L Output Signal Level	3	30	30	9	9	15	6	3
N Production Confidence Level		1.0	1.0	1.0	1.0	0.5	0.7	1.0
Total Without Confidence Level Factors Included		319	400	312	386	442	418	341
Final Figure of Merit		319	400	312	386	221	293	341

Organizational approach 2a does not require devices capable of NDRO operation. Therefore all components which have this capability (indicated in Table 8 by a Y) use a speed weight of 9, while those that do not have this capability (indicated by an N) use a speed weight of 7. Although, in the diode-capacitor (Dicap) memory, information is not completely destroyed upon interrogation, it does have to be rewritten frequently in order to prevent eventual loss. For this reason a speed weight of 7 was used for this device.

The highest rated components for approach 2a are plated wire and planar thin magnetic film devices (Table 4). Integrated circuits, the ferrite sheet memory, and biax elements run a fairly close second. The biax memory will probably not remain competitive much longer, since the parameters of the other devices are improving rapidly as new production techniques are developed. If the research and development problems associated with the cryotron and ferroelectric devices can be solved such that their confidence levels can be increased to 1.0, they would also fit into this second highest rated category.

The integrated circuit devices have the largest figure of merit for organizational approach 3 (Table 5). Most of the other components however have high ratings before the confidence levels are applied. Plated wire requires only a demonstration of signal cancellation logic capability to be highly rated. Attainment of this capability is doubtful in the case of planar magnetic films. The ability to produce workable cryotrons and thin-film transistor circuits on a large scale is the only factor holding the figure of merit down for these elements. The ferroelectrics and the ferrotron require an enhancement of both the production confidence level and the logic capability confidence level before they can be seriously considered for this approach.

Only the semiconductor type devices and the cryotron can be considered for approach 5 (Table 6). Before the confidence levels are taken into account, their figures of merit are all approximately equal. The same comments apply to the thin film transistor circuits and the cryotron as in the previous approach.

An entirely different class of components was considered for approach 2c, although some of them employ similar basic elements such as semiconductors or magnetic devices. The strain wave and the field-effect integrated circuit shift register devices have the largest figures of merit before the confidence level factors are applied and the field-effect device is on top after they are applied (Table 7). Considerable developmental effort is required on the strain wave devices before they can be considered for use in memory systems.

BLEM AREAS

Some of the devices previously discussed require further development effort in order to prove feasibility or to determine logic capability. Other devices, even though they already have large figures of merit, require additional effort to correct certain deficiencies in their characteristics. All of these problem areas are discussed in the following paragraphs.

Thin Magnetic Film Memories

Thin magnetic films of both the planar and the cylindrical (plated wire) configuration can be used as components of an associative memory cell. A problem that is common to both configurations concerns the non-destructive readout capability. Thin magnetic film memories are read by rotating the magnetization of the film, by means of a transverse magnetic field, toward the hard direction axis. The drive field magnitude must be such that the magnetization does not rotate the full 90 degrees, for if it does the information

stored in the memory will be destroyed since the magnetization vectors will split, half of them returning to one of the easy directions and half to the other easy direction. In an ideal thin magnetic film, the magnetization could be rotated almost 90 degrees, and, when the drive field is released, the magnetization would return to whichever easy direction it was originally oriented. In practical films, however, when the magnetization is rotated beyond a certain point, portions of the film return to the opposite direction when the drive field is removed. Thus, for non-destructive readout, the magnitude of the drive field which can be employed is limited, and as a result the output voltage on the sense line is also limited. Recently, coupled films consisting of two magnetic layers separated by a thin non-magnetic intermediate metallic layer have been investigated and they appear to be capable of NDRO operation with much larger transverse drive fields than is permissible with conventional single layer films. One of the two films in the sandwich has a low coercive force while the second film has a high coercive force. Additional research in this area is desirable to improve the NDRO capability of thin magnetic films thereby facilitating their use in those associative memory approaches which need this capability.

Cylindrical magnetic films (plated wire memories) can be made much thicker than those in the planar configuration because of the closed-flux nature of plated wire. This results in larger output voltages for the plated wire, which in turn increases the confidence in a signal cancellation logic capability. Several problems, however, still remain with plated wire. First, present wire sizes require larger drive currents than do planar films. This is a particularly severe problem in those types of associative memories where many drivers must be turned on simultaneously. Additional research to find methods of reducing these currents is essential. If they can be reduced sufficiently, integrated circuit drivers could be used which would in turn lower the cost, increase the reliability, and decrease the size of the memory. Another problem that has been observed is the aging effect. After plated wire has been stored for a period of time, the characteristics sometimes differ from those measured immediately after plating. Additional research is required to eliminate this effect.

isibility of performing signal cancellation logic should also be studied.
: of film uniformity, methods of standardizing drive currents, noise
ms, and unusual circuit techniques should be considered in this study
m.

r Integrated Circuits

been shown that integrated circuits employing bipolar transistors can be
) mechanize quite complicated associative memory cells. The two
al problems encountered in the use of bipolar integrated circuits in an
ative memory consist of (1) the large amount of power required and
potential reliability problem due to the necessity of interconnecting
ated circuit chips. The power consumption problem may be alleviated
use of pulse logic whereever possible, along with the use of circuits
tely employing PNP and NPN transistors. The circuits might be
ed in such a way that only the storage device or flip-flop is conducting
iable current except when the cell is being read into, read out of,
articipating in a logic operation. By means of these techniques, it
be possible to considerably reduce the average power consumption.

oblem of interconnecting integrated circuit chips has received con-
ble attention. First of all, the necessity of making these interconnec-
an be alleviated to some extent by making the chips larger and larger.
rse, the yield of operable circuits is the limiting factor in this approach.
be possible to increase the yield presently achieved with larger chips
ins of research by the semiconductor manufacturer. The second
l is increasing the reliability of large integrated circuit arrays, and
it has been studied at Honeywell, both under company funds and under
y contract funding, consists of making the interconnections by means
ecular bonds which eliminate intermetallic interfaces. These inter-
ire the most probable cause of interconnection failures.

Field Effect Integrated Circuits

Metal oxide semiconductor (MOS) field effect integrated circuits are subject to the same problems as bipolar integrated circuits and in addition they have the problem of a somewhat lower maximum operating speed. The relatively low limit on the maximum drain current in conjunction with the transistor and interconnecting lead capacity is the principal factor limiting the circuit speed. Device cutoff speeds are no longer an important factor. This problem can be alleviated by the use of complementary symmetry switching circuits which decrease the capacitive charge and discharge times. Industry efforts appear to be concentrated in this area, and recently published results (Ref. 6) indicate that delay times for MOS complementary symmetry memory cells are no more than 2 to 3 times those obtainable with conventional bipolar integrated circuits.

Cryotrons

The cryotron is a potentially ideal element for large associative memories of a size such that the power and size penalty which must be paid for the necessity of including a refrigerator can be divided among a large number of memory cells. The cells themselves should be inexpensive in large quantities because they are batch fabricated. Unfortunately, it has not been possible to construct large cryotron arrays due to a number of manufacturing problems. Some of these problems have been overcome, but many of them still remain, and there is some doubt that they will ever be satisfactorily solved.

Ferroelectrics

The ferroelectric and ferrotron devices require materials, fabrication, and circuit studies. Several new ferroelectric materials are presently being

investigated, and they appear quite promising. The ferroelectric elements are low-power devices, but methods must be developed for batch fabricating thin, small-area devices before these power levels can be achieved. Methods of performing logic must also be investigated.

Speed is the principal problem of the ferrotron. The speeds of presently usable photoconductors are two or three orders of magnitude below what is required for most associative memory applications. A potential advantage of the ferrotron is its optical input capability.

Ferroelectric materials might also be considered as emergency storage elements within an integrated circuit memory cell, as suggested in Reference 1. Ferroelectrics are much better than any other non-volatile device for this application since their drive requirements and output voltages most nearly match the corresponding characteristics of semiconductors. In this application the ferroelectric is set by the integrated circuit as soon as incipient power failure is sensed. When power is restored all data is automatically transferred back into the flip-flops.

RECOMMENDED AREAS FOR FURTHER DEVELOPMENT

Of the devices described above a number have been selected as the most fruitful for research projects for the following reasons:

- The devices have a high figure of merit in one or more of the approaches or would have if the problems involved can be solved.
- The problem areas do not appear to be receiving sufficient attention in industry at the present time.
- The problem areas appear to be capable of solution with reasonable effort.

The suggested research projects for each of the selected devices are:

(1) Planar Magnetic Thin Film -- Investigate use of coupled films to improve NDRO Capability.

(2) Plated Wire --

(a) Same as for planar films

(b) Use of smaller wire to reduce drive current requirements

(c) Solution of aging problems

(d) Investigate techniques for performing logic with plated wire.

(3) Bipolar Integrated Circuits --

(a) Investigate methods of improving speed power ratio by the use of techniques such as pulse logic employing alternate NPN and PNP stages and complementary symmetry flip-flops.

(b) Study methods of making integrated circuit memory cells non-volatile by the use of ferroelectric materials.

(c) Investigate more reliable methods of interconnecting integrated circuit chips.

A number of other devices which had high figures of merit for one or more of the organizational approaches, but were excluded either because they are presently receiving a lot of attention or because the amount of research required to solve the problems may be unreasonable are listed below. In some cases emphasis on a particular characteristic could make research programs on one of these devices justifiable. The suggested area of research in each case is also listed.

Thin-film transistors -- Materials and fabrication research is required.

Field effect integrated circuits -- Research should be aimed at increasing the speed.

Strain wave -- Materials and fabrication studies are required.

Ferroelectric devices -- Research on materials and batch fabrication techniques is needed.

Ferrotron -- Materials studies and research aimed at increasing the speed are required. A requirement for optical input could make this device more desirable than its figures of merit would indicate.

SECTION V
CONCLUSIONS AND RECOMMENDATIONS

CONCLUSIONS

Probably the most significant single conclusion drawn from this work is that associative memories and associative processors will be important additions to the computer system on space vehicles.

Two applications in particular help justify this conclusion. One of these is concerned with the collection of data of both the scientific and system status types. The associative processor proposed for this application functions both as a data compressor and as a readily programmable multiplexer. The search capabilities are used to select a subset of sensors for a given situation and the processing capabilities allow an approach to data compression in which the outputs of all sensors are examined jointly to determine which has the most significant information.

The advantages of this approach to data collection can be summarized as follows:

- The most pertinent data is selected each sampling cycle.
- The data gathering rate can be matched to the current rate of the communications channel.
- The associative memory also provides adaptive selection of sensors.
- Associations between sensors can readily be made by assigning special bits over which searches can be performed. This would allow the acquisition of data from a group of sensors whenever one of them, for instance, is found to have pertinent data.

it is concluded that an associative processor can provide a valuable
bility in this data acquisition, data compression problem area.

other application area where associative memories are expected to make
nificant contribution is in multi-processor systems. It has been shown
he control function in a flexible and expandable multi-processor is of
ient complexity that hardware approaches should be considered. Func-
such as task assignment and memory allocation involve table lookup
itions of such a type that associative storage of the table provides major
itages. In addition, an associative multi-access memory in which words
ccessed serially eliminates the central switch normally used in multi-
essors. A multi-processor using the distributed version of this memory,
ich the access circuitry is distributed throughout the memory array,
ome interesting characteristics. It has reliability advantages since the
ory and switch functions are reduced to a single cellular array in which
es should affect only a single word of memory or a single processor.
readily expandable since the connections between processors and the
ory switch array are the same regardless of the number of words of
ory. It also provides a possible advantage in parallel processing since
erial processors operate one bit time out of synchronism with one another.

other applications studied, while not as attractive as the first two, are
e processing and incremental computations. Picture processing was
tigated by picking a particular cloud processing problem in which the
er of clouds in the binary representation of the picture and the number
in each cloud was desired. An attempt to apply two-directional search
ilities (searching over both rows or columns of bits) to this problem
nsuccessful. A bit slice processing mode of operation, where a word
sociative processor was provided for each row of the picture, turned
be the most reasonable approach. The parallel processing capabilities
associative processor provide extreme flexibility since a scratchpad
n of each word can serve as a counter and as an adder and can be
hed as well. In general, it is concluded that associative processors

can be applied effectively to picture processing type problems. However, due to the somewhat special nature of the specific problem examined, no other conclusions can be drawn.

The final application investigated was that of doing incremental computations in an associative processor in which each word serves as an integrator. Probably the most significant advantage that the associative capabilities provide is the flexibility in interconnecting these integrators. The plugboard usually required in such systems is eliminated by performing searches on special fields of each word. An interesting observation concerning the requirements for incremental computations is that they occur primarily during the landing phase. The significance of this fact is that an associative processor, justified mainly for some other function, could handle this requirement during its relatively short duration and thus eliminate the special-purpose hardware which would normally be furnished.

Other conclusions that can be drawn from this study are in the area of mechanization of associative memories for space applications. In addition to the four applications summarized above, several others were considered in less detail, and the associative memory organizational approaches suitable for each were specified. From this exercise, it became clear that two approaches stand out. One of these, which was named approach 2a in Task 2, has no logic distributed throughout the memory array, has a bit slice writing capability, and has an external serial adder per word. The other, called approach 3, has distributed logic to the extent that an all-parallel equality search can be performed. Two other approaches, although of less importance than the first two, were also required for one or more of the applications considered. These are approach 2c and approach 5. The main features of 2c are that it has no distributed logic and allows cyclic access to a bit slice, rather than random access as in all the other approaches. Approach 5 is a powerful associative processor with complex intercommunicating cells.

e evaluation of devices for each of these approaches is described in detail Section IV. Planar magnetic thin films, plated wire, and bipolar integrated circuits are devices at which specific development recommendations e aimed.

IMENDATIONS

a result of the associative memory application investigations and mechaation evaluations performed on this study, a number of areas in need of ther effort were identified. These are described below:

- It is recommended that the adaptive data acquisition application be given further attention. The system design and simulation of an associative adaptive data acquisition system should be performed to determine those functions it can perform and its effectiveness on each. It is also recommended that the development of an associative processor of the type suitable for this application be initiated.

- The feasibility of a multi-access associative memory should be given more study. System design of a multi-processor system using such a memory should also be carried out. The specific control and memory functions to be performed by the multi-access memory should be defined.

- Since organizational approach 2a and 3 are the most frequently used in space applications it is recommended that mechanization efforts be directed at their requirements.

- Recommended research and development programs aimed at the improvement of devices for use in associative memories are as follows:

(a) Planar Magnetic Thin Films - Investigate coupled films

(b) Plated Wire

(1) Investigate coupled films

(2) Investigate the use of smaller wire to reduce drive current requirements

(3) Find solution to the aging problem

(4) Investigate techniques for performing logic with plated wire

(c) Bipolar Integrated Circuits

(1) Investigate methods of improving the speed power ratio by the use of techniques such as pulse logic employing alternate NPN and PNP stages and complementary symmetry flip-flops.

(2) Study methods of making integrated circuit memory cells non-volatile by the use of ferroelectric materials

(3) Investigate more reliable methods of interconnecting integrated circuit chips.

REFERENCES

D. C. Gunderson, C. W. Hastings, G. J. Prom, "Spaceborne Memory Organization", Interim Report for Contract NAS 12-38, Honeywell Systems and Research Division, 15 Dec 1965.

W. R. Bechfold, J. E. Medlin, D. R. Weker, Final Report - PCM Telemetry Data Compression Study, Phase I, Lockheed Missiles and Space Company, contract NAS 5-9729, October 1965.

P. Drapkin and R. M. Pentz, "An Associative Data Compressor," IEEE International Convention Record - Part I, 22-26 March 1965, pages 231-236.

Analog Digital Conversion Handbook, Publication No. E-5100, 1064, Digital Equipment Corporation.

Data Compression Techniques Study, Honeywell Aeronautical Division proposal to Goddard Space Flight Center (NASA), 14 May 1964.

L. Bloom, M. Cohen, and S. Porter, "Considerations in the Design of a Computer with a High Logic-to-Memory Speed Ratio", Proceedings on Gigacycle Computing Systems, 29 Jan. - 2 Feb. 1962.

"Summary of Investigation on Associative Memories", Computer Command and Control Company, Contract No. 4068(00)ONR, 15 Jan. 1964.

Burns, J. R. et al, "Integrated Memory using Complementary Field Effect Transistors", Digest of Technical Papers, 1966 International Solid State Circuits Conference, Philadelphia, Pa.

D. R. Weber, "A Synopsis on Data Compression", Proceedings of the 1965 National Telemetering Conference, April 1965, pages 9-16.

"The Use of Data Omission for Redundancy Removal", C. J. Palermo, and H. Horwitz, technical memo, Institute of Science and Technology, University of Michigan, late 1965.

F. W. Kantor, "Telemetry Concept Could Speed Space Program", Electronics, 12 Oct. 1962, Pages 60-62.

APPENDIX A
INTERPOLATOR AND PREDICTOR ALGORITHMS

APPENDIX A
INTERPOLATOR AND PREDICTOR ALGORITHMS

The operation of corridor-type interpolator algorithms will be illustrated for the case of zero-order and first-order interpolators, for a single data source monitoring a physical variable y(t). A description of several predictor algorithms then concludes this appendix. References 2 and 9 are recommended for additional information.

THE ZERO-ORDER INTERPOLATOR ALGORITHM

In the zero-order interpolator algorithm (ZOI in the system of mnemonics of Reference 2), the first sample $y(t_1)$ received after a data point $y^*(t_0)$ has been transmitted is taken as the "anchor point" for the ensuing sequence of data points. (The * symbol here signifies that $y^*(t_0)$ is generally not a real data sample, but a computed value.) Assuming that the tolerance ϵ has been assigned to this data source, a "corridor upper bound" $u(t_1)$ is taken as $y(t_1) + \epsilon$ and a "corridor lower bound" $\ell(t_1)$ is taken as $y(t_1) - \epsilon$. When $y(t_2)$ is acquired, $u(t_1)$ and $\ell(t_1)$ are compared with $y(t_2)$. There are five possible cases:

$$\left.\begin{array}{ll} \text{(a)} & y(t_2) < \ell(t_1) - \epsilon \\ \text{(b)} & \ell(t_1) - \epsilon \le y(t_2) < y(t_1). \\ \text{(c)} & y(t_2) = y(t_1). \dagger \\ \text{(d)} & y(t_1) < y(t_2) \le u(t_2) + \epsilon. \\ \text{(e)} & u(t_2) + \epsilon < y(t_2). \end{array}\right\} \quad \text{(A.1)}$$

† In a digital comparison, this case has an appreciable probability of occurrence.

These possible cases are illustrated in Figure A-1. Note that cases (b) and (d) may include a new data point which falls either within or outside the corridor. The corridor specifies, at any given time, the range of possible positions of the line segment which finally will be chosen to represent the sequence of data now being acquired, there is not a direct specification that all subsequent data points of the sequence must fall within the corridor or else the sequence will terminate. (Such a specification is made here only in the case of the "ordinary fan" algorithm, considered in the next subsection). The action taken in each case is as follows:

$$\left.\begin{array}{ll}
\text{(a)} & \text{Transmit } y(t_1). \text{ Begin a new sequence with } y(t_2) \text{ as anchor point.} \\
\text{(b)} & \text{Continue sequence. } u(t_2) = y(t_2) + \varepsilon,\ \ell(t_2) = \ell(t_1). \\
\text{(c)} & \text{Continue sequence. } u(t_2) = u(t_1),\ \ell(t_2) = \ell(t_1). \\
\text{(d)} & \text{Continue sequence. } u(t_2) = u(t_1),\ \ell(t_2) = y(t_2) - \varepsilon. \\
\text{(e)} & \text{Transmit } y(t_1). \text{ Begin a new sequence with } y(t_2) \text{ as anchor point.}
\end{array}\right\} \quad \text{(A.2)}$$

Initially, $u(t_1) - \varepsilon = y_1(t) = \ell(t_1) + \varepsilon$, that is, $u(t_1) - \ell(t_1) = 2\varepsilon$. After a point $y(t_k) \neq y(t_1)$ has been sampled, the difference $u(t_k) - \ell(t_k)$ becomes and remains less than 2e; it may be seen that this difference is a monotonic non-increasing function of t. The general situation after k pieces of data have been sampled is like that for k = 1, except that case (c) must be defined more generally as meaning that $y(t_{k+1})$ falls within ε of both $u(t_k)$ and $\ell(t_k)$. The value to be transmitted, when that is necessary, is chosen as the midway point between $\ell(t_k)$ and $u(t_k)$, that is:

$$m(t_k) = \ell(t_k) + \frac{u(t_k) - \ell(t_k)}{2} = \frac{u(t_k) + \ell(t_k)}{2} \qquad \text{(A.3)}$$

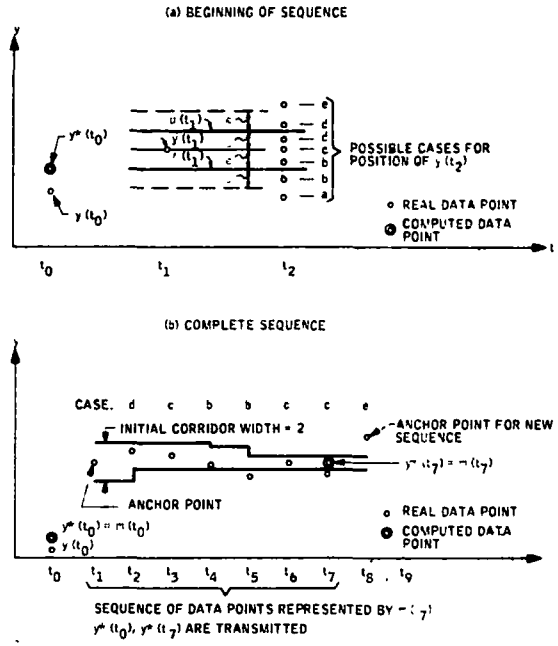


Figure A1. Operation of ZOI Algorithm

The possible cases are:

$$\left.\begin{array}{ll} \text{(a)} & y(t_{k+1}) < \ell(t_k) - \epsilon \\ \text{(b)} & \ell(t_k) - \epsilon \le y(t_{k+1}) < u(t_k) - \epsilon \\ \text{(c)} & u(t_k) - \epsilon \le y(t_{k+1}) \le \ell(t_k) + \epsilon \\ \text{(d)} & \ell(t_k) + \epsilon < y(t_{k+1}) \le u(t_k) + \epsilon \\ \text{(e)} & u(t_k) + \epsilon < y(t_{k+1}). \end{array}\right\} \quad \text{(A.4)}$$

The action taken in each case is as follows:

$$\left.\begin{array}{ll} \text{(a)} & \text{Transmit } m(t_k). \text{ Begin a new sequence with } y(t_{k+1}) \text{ as anchor point.} \\ \text{(b)} & \text{Continue sequence. } u(t_{k+1}) = y(t_{k+1}) + \epsilon, \ \ell(t_{k+1}) = \ell(t_k) \\ \text{(c)} & \text{Continue sequence. } u(t_{k+1}) = u(t_k), \ \ell(t_{k+1}) = \ell(t_k) \\ \text{(d)} & \text{Continue sequence. } u(t_{k+1}) = u(t_k), \ \ell(t_{k+1}) = y(t_{k+1}) - \epsilon \\ \text{(e)} & \text{Transmit } m(t_k). \text{ Begin a new sequence with } y(t_{k+1}) \text{ as anchor point.} \end{array}\right\} \quad \text{(A.5)}$$

Figure A-1(b) illustrates the behavior of this algorithm for a few samples, comprising somewhat more than one sequence. Observe that the "computed values" $y^*(t_0)$ has now been identified as $m(t_0)$, corresponding to a preceding sequence of data such that $y(t_1)$ was the first point to fall in case (a) or case (e).

ST-ORDER INTERPOLATOR ALGORITHMS

'st-order polynomial interpolator may be defined in an analogous manner. ever, there is more than one possible first-order interpolator algorithm, nding on the choice of parameters to determine a straight-line segment. all first-order interpolators can be implemented as memory-saving corridor- algorithms similar to the one just described. One discussed here is called "first-order interpolator joined line segment" (mnemonic FOIJON). Unlike OI algorithm, the FOIJON algorithm takes the last data value $y^*(t_0)$ trans- ed as the anchor point for a new sequence; $y^*(t_0)$ also defines the end-point e preceding line segment. $u(t_1)$ is now no longer a horizontal line deter- d by a constant, rather, it is a sloping line determined initially by the irement that it pass through $y^*(t_0)$ and $y(t_1) + \epsilon$. $\ell(t_1)$ is likewise specified as loping line passing through $y^*(t_0)$ and $y(t_1) - \epsilon$. Let $\alpha(t_k)$ be the slope of $u(t_k)$ etermined by the algorithm at time t_k; likewise $\beta(t_k)$ is the slope of $\ell(t_k)$ etermined by the algorithm at time t_k; likewise $\beta(t_k)$ is the slope of $\ell(t_k)$ at t_k. Then

$$\left.\begin{aligned} y(t_1) + \epsilon &= y^*(t_0) + \alpha(t_1)\,(t_1 - t_0) \\ y(t_1) - \epsilon &= y^*(t_0) + \beta(t_1)\,(t_1 - t_0) \end{aligned}\right\} \quad \text{(A.6)}$$

ing these for $\alpha(t_1)$ and $\beta(t_1)$,

$$\left.\begin{aligned} \alpha(t_1) &= \frac{y(t_1) + \epsilon - y^*(t_0)}{t_1 - t_0} \\ \beta(t_1) &= \frac{y(t_1) - \epsilon - y^*(t_0)}{t_1 - t_0} \end{aligned}\right\} \quad \text{(A.7)}$$

The extrapolated corridor limit values of y(t) at time t_{k+1}, corresponding to y(t) lying on $u(t_k)$ and on $\ell(t_k)$ respectively, are given by:

$$\left.\begin{aligned} y_u(t_k) &= y^*(t_0) + \alpha(t_k)\ (t_{k+1} - t_0) \\ y_\ell(t_k) &= y^*(t_0) + \beta(t_k)\ (t_{k+1} - t_0) \end{aligned}\right\} \quad (A.8)$$

The notations $u(t_k)$ and $\ell(t_k)$ do not denote fixed values in the discussion of this section, although they did in the preceding section for each value of k. The fixed values are $y_u(t_{k+1})$ and $y_\ell(t_{k+1})$. Equation (A.8) is easily seen to hold for $k \geq 1$. The notation $\ell(t_k) - \epsilon$, $\ell(t_k) + \epsilon$ similarly denote sloping lines parallel to $\ell(t_k)$, and so forth. No comparison is done for $y(t_1)$. When $y(t_2)$ is acquired, there are again five possible cases:

$$\left.\begin{aligned} &\text{(a)}\ y(t_2) \text{ is below } \ell(t_1) - \epsilon, \text{ that is, } y(t_2) < y_\ell(t_2) - \epsilon \\ &\text{(b)}\ y_\ell(t_2) - \epsilon \leq y(t_2) < y_\ell(t_2) + \epsilon \\ &\text{(c)}\ y_\ell(t_2) + \epsilon \leq y(t_2) \leq y_u(t_2) - \epsilon \\ &\text{(d)}\ y_u(t_2) - \epsilon < y(t_2) \leq y_u(t_2) + \epsilon \\ &\text{(e)}\ y_u(t_2) + \epsilon < y(t_2). \end{aligned}\right\} \quad (A.9)$$

These possible cases are illustrated in Figure A-2(a). As before, cases (b) and (d) may include a new data point falling within or outside the corridor. The action taken in each case is as follows:

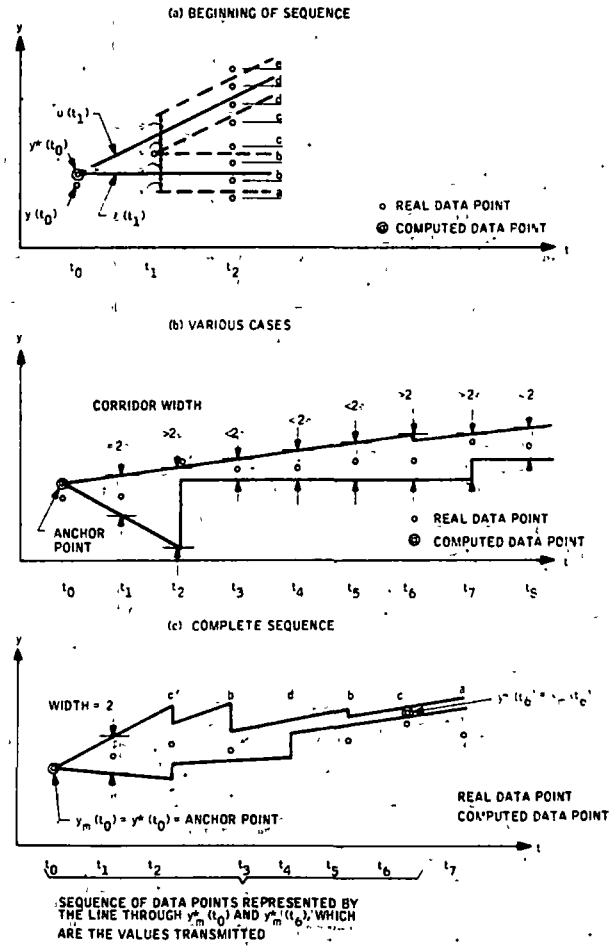


Figure A2. Operation of FOLJON Algorithm

(a) Transmit $y(t_1)$. Begin a new sequence with $y(t_1)$ as anchor point.

(b) Continue sequence. $\alpha(t_2)$ is now determined so that $u(t_2)$ passes through $y(t_2) + \epsilon$, that is, $\alpha(t_2) = \dfrac{y(t_2) + \epsilon - y^*(t_0)}{t_2 - t_0}$. $\ell(t_1)$ does not need to be changed, so $\beta(t_2) = \beta(t_1)$.

(c) Continue sequence. Both $u(t_1)$ and $\ell(t_1)$ need to be changed, so $\alpha(t_2)$ and $\beta(t_2)$ are determined so that $u(t_2)$ passes through $y(t_2) + \epsilon$, and $\ell(t_2)$ passes through $y(t_2) - \epsilon$; that is,

$$\alpha(t_2) = \frac{y(t_2) + \epsilon - y^*(t_0)}{t_2 - t_0}, \text{ and } \beta(t_2) = \frac{y(t_2) - \epsilon - y^*(t_0)}{t_2 - t_0}.$$

(d) Continue sequence. $u(t_1)$ does not need to be changed, so $\alpha(t_2) = \alpha(t_1)$. $\beta(t_2)$ is now determined so that $\ell(t_2)$ passes through $y(t_2) - \epsilon$, that is, $\beta(t_2) = \dfrac{y(t_2) - \epsilon - y^*(t_0)}{t_2 - t_0}$.

(e) Transmit $y(t_1)$. Begin a new sequence with $y(t_1)$ as anchor point. (A.10)

The general situation is more complicated for the FOIJON algorithm than for the ZOI algorithm, in that at time t_{k+1} the corridor width $y_u(t_{k+1}) - y_\ell(t_{k+1})$ may be either greater or less than 2ϵ, since $u(t_k)$ and $\ell(t_k)$ are always divergent from each other. As k increases, this width cannot exceed 2ϵ by much, and it may become less than 2ϵ; but it can still exceed 2ϵ by a small amount even for k fairly large. The situation in this event is still basically that of Figure A-1(a); although the angle between $u(t_k)$ and $\ell(t_k)$ will be very small compared to that between $u(t_1)$ and $\ell(t_1)$.

corridor width is small, a different situation is possible in which $y(t_{k+1})$ e within ϵ of $u(t_k)$ and $\ell(t_k)$ instead of farther than ϵ from either. = A-2(b) illustrates this situation. Thus, a new data point in the middle corridor may mean that both u(t) and $\ell(t)$ have to be changed, or that r of them do; the first possibility has already been labeled case (c), so cond is called case (c'). The value to be transmitted, when that is sary, is chosen as $y_m(t_k)$ where $m(t_k)$ is the line halfway between the lines ind $\ell(t_k)$; that is

$$m(t_k) = y^*(t_0) + \left(\frac{\alpha(t_k) + \beta(t_k)}{2}\right)(t_k - t_0) \qquad \text{(A.11)}$$

ossible cases are:

a) $y(t_{k+1}) < y_\ell(t_{k+1}) - \epsilon$

b) $y_\ell(t_{k+1}) - \epsilon \le y(t_{k+1}) < \min\left\{y_\ell(t_{k+1}) + \epsilon,\ y_u(t_{k+1}) - \epsilon\right\}$

c) $y_\ell(t_{k+1}) + \epsilon \le y(t_{k+1}) \le y_u(t_{k+1}) - \epsilon$

c') $y_u(t_{k+1}) - \epsilon \le y(t_{k+1}) \le y_\ell(t_{k+1}) + \epsilon$

d) $\max\left(y_u(t_{k+1}) - \epsilon,\ y_\ell(t_{k+1}) + \epsilon\right) < y(t_{k+1}) \le y_u(t_{k+1}) + \epsilon$

e) $y_u(t_{k+1}) + \epsilon < y(t_{k+1})$. (A.12)

event that $y(t_{k+1}) = y_\ell(t_{k+1}) + \epsilon = y_u(t_{k+1}) - \epsilon$, there would be no need compute the slope of $\ell(t)$ or u(t) until $y(t_{k+2})$ was received. Hence, this ole situation is assigned to case (c').

The action taken in each case is as follows:

(a) Transmit $y_m(t_k)$. Begin a new sequence with $y_m(t_k)$ as anchor point.

(b) Continue sequence. $\alpha(t_{k+1})$ is now determined so that $u(t_{k+1})$ passes through $y(t_{k+1}) + \epsilon$, that is, $\alpha(t_{k+1}) = \dfrac{y(t_{k+1}) + \epsilon - y^*(t_0)}{t_{k+1} - t_0}$. $\ell(t_k)$ does not need to be changed, so $\beta(t_{k+1}) = \beta(t_k)$.

(c) Continue sequence. Both $u(t_k)$ and $\ell(t_k)$ need to be changed, so $\alpha(t_{k+1})$ and $\beta(t_{k+1})$ are determined so that $u(t_{k+1})$ passes through $y(t_{k+1}) + \epsilon$, and $\ell(t_{k+1})$ passes through $y(t_{k+1}) - \epsilon$, that is, $\alpha(t_{k+1}) = \dfrac{y(t_{k+1}) + \epsilon - y^*(t_0)}{t_{k+1} - t_0}$ and $\beta(t_{k+1}) = \dfrac{y(t_{k+1}) - \epsilon - y^*(t_0)}{t_{k+1} - t_0}$. This case can occur only if $y_u(t_{k+1}) - y_\ell(t_{k+1}) > 2\epsilon$.

(c') Continue sequence. Neither $u(t_k)$ nor $\ell(t_k)$ need to be changed, so $\alpha(t_{k+1}) = \alpha(t_k)$ and $\beta(t_{k+1}) = \beta(t_k)$. This vase can occur only if $y_u(t_{k+1}) - y_\ell(t_{k+1}) \leq 2\epsilon$.

(d) Continue sequence. $u(t_k)$ does not need to be changed, so $\alpha(t_{k+1}) = \alpha(t_k)$. $\beta(t_k)$ is now determined so that $\ell(t_{k+1})$ passes through $y(t_{k+1}) - \epsilon$, that is $\beta(t_{k+1}) = \dfrac{y(t_{k+1}) - \epsilon - y^*(t_0)}{t_{k+1} - t_0}$.

(e) Transmit $y_m(t_k)$. Begin a new sequence with $y_m(t_k)$ anchor point.

Figure A-2(c) illustrates the behavior of this algorithm for a few samples, comprising somewhat more than one sequence. Observe that the "computed value" $y^*(t_0)$ has now been identified as $y_m(t_0)$, corresponding to a preceding sequence of data such that $y(t_{-1})$ was the first point to fall in case (a) or (e).

IJON algorithm is economical because it requires only one new data define each new line segment (after the first segment, of course). It owever, have the property -- as does the ZOI algorithm also -- that a points do not normally get sent as part of the broken-line-segment curve the real data. The incorporation of an occasional real data point in smitted data improves the fidelity of reconstruction, moreover, back-and-ershoot or "oscillation" of the fitted line about the real data is possible of the line segments terminate in real data points. A modified first-nterpolator which does incorporate real data points is the "first-order ator disjoined line segment" (mnemonic FOIDIS). This algorithm A-3) differs from the FOIJON algorithm only in that the transmitted end a preceding line segment, $y_m(t_k)$, is no longer the anchor point for the e segment, rather $y(t_{k+1})$ is chosen as this anchor point. Although the algorithm incurs the immediate disadvantage that twice as much data wo separate points) must be transmitted to define each line segment, rical performance was nevertheless found to be excellent by the authors ence 2, the choice of a real data point as an anchor point seemed to de- (A.13)
he tendency of the algorithm to oscillate, to the point where sequences d generally more than twice as long as with the FOIJON algorithm "under nditions". In fact, the FOIDIS algorithm was by implication the first ltogether in the evaluation of Reference 2, it may as well be used in studies of the type under consideration here, since it can be programmed ssociative processor as readily as can the FOIJON or ZOI algorithms.

er first-order interpolator discussed in Reference 2, the "four degree om" interpolator (mnemonic FOI4DG), is not particularly suitable for ive processor implementation. The reason is that this algorithm (Figure quires the retention of all of the data points of a sequence until the e is complete, which means that the memory allocation for each word in ciative processor must be "open-ended" - that is, the computer would ow at the beginning of a sequence exactly how many data fields of storage it g to need. No anchor point is used for the FOI4DG algorithm. Instead,

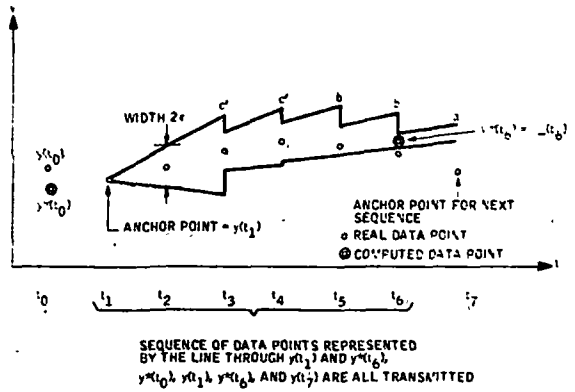


Figure A3. Operation of FOIDIS Algorithm - Complete Sequence

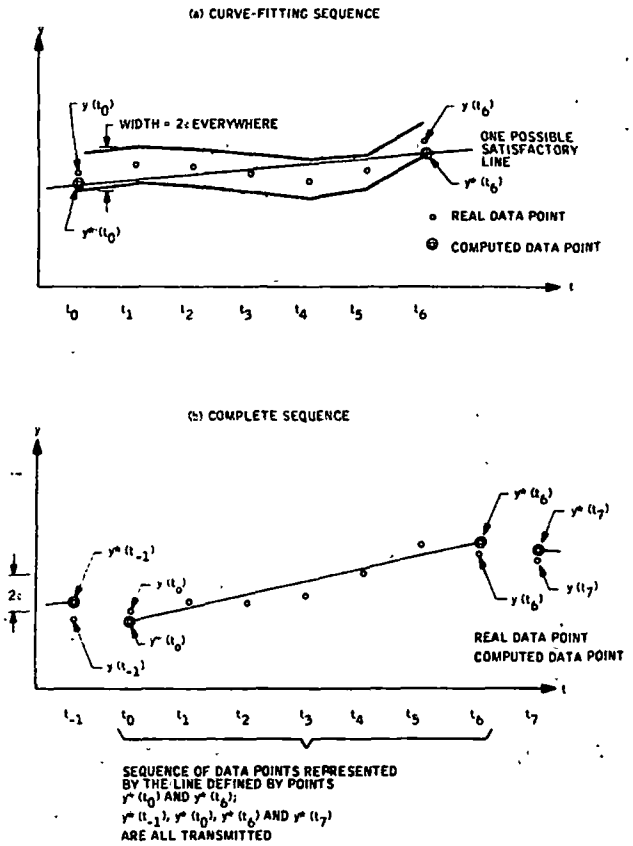


Figure A4. Operation of FOI4DG Algorithm

a straight-line curve-fit is made to the points of the sequence, and recomputed for each new point, by any computational technique which has the property of minimizing the maximum error of this curve-fit, or at least of doing so for practical purposes. (The formal mathematical term for a curve-fit which minimizes the maximum error is the "best fit in the Tchebycheff sense".) When this maximum error exceeds the allowed tolerance, the sequence is terminated and transmission takes place. Graphically, the situation is as shown in Figure A-4(a); Figure A-4(b) illustrates the behavior of this algorithm for a few samples comprising somewhat more than one sequence. All of the points of the sequence are now required in order to define a "corridor" (now with irregular sides). The sequence must be terminated whenever it is no longer possible to pass any straight line through this "corridor" without crossing the "corridor" boundary at some point. The virtue of the FOI4DG algorithm is that the length of each line segment is maximized; from a theoretical point of view it is the best possible first-order interpolator according to the usual requirement that the data be guaranteed accurate to some given tolerance. (This sort of guarantee normally implies the Tchebycheff criterion.) From a practical point of view, however, the FOIDIS algorithm is preferable since it gives almost as good a curve-fit, the computational problem it poses is vastly simpler, and only the anchor point and the two line slopes need to be retained in storage regardless of the number of points in the sequence. Reference 10 also discusses the FOI4DG algorithm to be less effective than the "ordinary fan" (described next) on theoretical grounds.

The "fan" or "ordinary fan" method discussed in Reference 10 is another possible variation of the FOIJON algorithm. The line segments are still joined, but the anchor point chosen is always a real data point, hence only real data points are transmitted. The computational procedure given in the reference is to test, for each $y(t_{k+1})$, all preceding data points of the sequence to see if they are within ϵ of the straight line joining $y(t_0)$ (the anchor point) and $y(t_{k+1})$. This procedure as described requires an open-ended memory allocation, and hence is unsuitable for use in an associative processor. However, this drawback can easily be avoided; a suitable corridor-type algorithm is obtained by

taking a real data point $y(t_0)$ as anchor point, and iterating in the same way as for the FOIJON algorithm, but terminating the sequence when a point $y(t_{k+1})$ falls outside the corridor (not when a point falls more than ϵ outside the corridor as for the FOIJON algorithm). The straight line segment from $y(t_0)$ to $y(t_k)$ has the required property, and $y(t_k)$ is the anchor point for a new sequence. It would not seem as if the average length of a sequence for this algorithm would be as great as for the FOIJON algorithm, but on the other hand the use of real data points might be beneficial in reducing oscillation, and hence it is not obvious which of the two would be better - perhaps it depends rather empirically on the data.

Reference 10 also discusses a "fan method using conditional means". This algorithm will not be considered here, since it presupposes knowledge of the signal spectrum. Any such presupposition is at least in principle a fatal drawback for any method proposed for use in compressing data gathered from previously unexplored planets, since if a signal has never previously been sampled it is difficult to see how it can be mathematically treated as having a known spectrum.

REDICTOR ALGORITHMS

Predictor algorithms are somewhat simpler than interpolator algorithms of the same order. Reference 2 found them to be generally less effective than interpolator algorithms, on the basis of large-scale numerical experiments, although one predictor algorithm was still among the six best algorithms found.

An advantage cited for predictor algorithms over other data compression algorithms, which could be important in some aerospace applications, is that the major part of the computations can be performed at the receiver end instead of on board the vehicle (see Reference 11). This comes to pass as follows: On the basis of previously transmitted data, the receiving station predicts what the current reading will be, and transmits this predicted reading to the vehicle; the

vehicle then transmits only data which disagree with the prediction more than an allowable tolerance ε. This advantage is presently of questionable importance, since it is no longer exhorbitantly expensive to put an appreciable amount of computing power on board an aerospace vehicle. However, if the "system" were to consist of a large aerospace vehicle ("mother ship") which in turn made use of small instrumented data-gathering drones, the same reasoning might again legitimately apply; the "receiver" would now be aboard the vehicle, and the drone would probably have no significant computing capability, whereas the vehicle itself would have a fairly sophisticated data processing system.

The simplest type of prediction algorithm is the "simple difference" technique (Ref. 2, page I-5) which consists of just computing the difference of each new data point and the one preceding it, and transmitting a new data point whenever this difference exceeds a certain threshold ε; that is, $y(t_{k+1})$ is transmitted whenever $|y(t_{k+1}) - y(t_k)| > \epsilon$. This algorithm has the severe disadvantage that considerable inaccuracy can result during periods when the rate of change in the sensed variable is quite slow and the threshold is seldom exceeded. The algorithm may be useful, however, if for some reason the transmission criterion is to send a new data point whenever the first derivative of the variable with respect to time exceeds some threshold; in this event, if the sampling time interval is uniform the algorithm does exactly what it should. This algorithm is also referred to as the "step method" (Reference 10).

Another algorithm, referred to as the "drop near zero" method (Reference 10) consists of permanently picking a predicted value for the variable in question, and then suppressing transmission of all data points within ε of that predicted value. This method would obviously be highly inefficient with data that was at all interesting; it would lead to a transmitter overload condition whenever unexpected conditions were encountered on several sensors simultaneously.

lgorithm, given the mnemonic ZOP for "zero-order predictor" in Ref- (see page I-5 of this reference) since it is probably the simplest --type algorithm worth consideration for most applications, consists of g the "drop near zero" algorithm so that the predicted value is changed e a data point is transmitted. In particular, the new predicted value st out-of-tolerance data point encountered, the one which led to the ion of the preceding sequence. When a data point is observed which ı more than ε from this new predicted value, transmission occurs and quence is begun. The predicted value is the value actually transmitted, the beginning of the sequence (as indicated in Reference 2) or at the e sequence (as indicated in the sample sequence shown in Figure A-5). / important that ε be several times as great as the minimum binary n (the numeric value of a "one" in the least significant bit of a data order to avoid oscillation. This algorithm was one of the six which p best in the study of Reference 2.

t-order predictor algorithms were also investigated in this same study :rence 2, pages I-7 through I-13). Only one of these was found even :he performance of the ZOP algorithm, and it did not do so consistently. ıe authors of Reference 2 take a rather pessimistic view of the value .hms of this class, and also of predictors of second and higher order y could be expected to exhibit the same weaknesses as first-order 's.

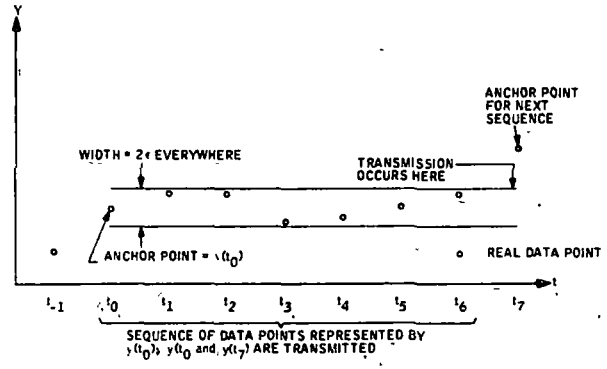


Figure A5. Operation of ZOP Algorithm - Complete Sequence