UNCLASSIFIED

AD NUMBER

AD489660

NEW LIMITATION CHANGE

TO

Approved for public release, distribution unlimited

FROM

Distribution authorized to U.S. Gov't. agencies and their contractors; Administrative/Operational Use; Jul 1996. Other requests shall be referred to Rome Air Development Center, Griffis AFB, NY.

AUTHORITY

RADC USAF ltr, 17 Sep 1971

THIS PAGE IS UNCLASSIFIED

489660

RADC-TR-65-397, Vol I
F'NAL REPORT

ANALYSIS OF SMALL ASSOCIATIVE MEMORIES FOR DATA STORAGE AND RETRIEVAL SYSTEMS

VOLUME I, MANAGEMENT REPORT

Robert S. Green, Dr. Jack Minker, and Warren E. Shindle

TECHNICAL REPORT NO. RADC-TR-65-397, Vol I

July 1966

This document is subject to special export controls and each transmittal to foreign governments or foreign nationals may be made only with prior approval of RADC (EMLI), GAFB, N.Y. 13440.

Rome Air Development Center
Research and Technology Division
Air Force Systems Command
Griffiss Air Force Base, New York

ANALYSIS OF SMALL ASSOCIATIVE MEMORIES FOR DATA STORAGE AND RETRIEVAL SYSTEMS

VOLUME I, MANAGEMENT REPORT

Robert S. Green, Dr. Jack Minker, and Warren E. Shindle

This document is subject to special export controls and each transmittal to foreign governments or foreign nationals may be made only with prior approval of RADC (E .LI), GAFB, N.Y. 13440.

MAINLINE ENGINEERING MAPLE SHADE, N.J.
9/26/66 169

FOREWORD

This report was prepared by the Auerbach Corporation, Philadelphia, Pennsylvania, under Contract Number AF30(602)-3564; Project Number 4594 and Task Number 459402. The RADC project engineer is Ronald J. Ferris, EMIID.

This is a final report and covers the period of work from October 1964 to September 1965; the originator's report number is 1231-TR2.

The results of this study are contained in two volumes:
Volume I presents in summary form, an introduction, findings, conclusions, and recommendations. In addition, this volume contains a discussion on associative memories and the comparative analyses of the associative memory configurations studied. A short critique of the Goodyear Associative Processor and a bibliography are also given in this volume.

Volume II contains the detailed discussion of the study. This volume specifies the problem of "how" the study was conducted. It presents the technical details, the relevant flowcharts, algorithms, and encoding. Each section of this volume contains information pertinent to the approach, technical details, and the findings and conclusion. Supplementary information relevant to the study is contained in the appendices.

The principal investigators on this study were Mr. James Dugan, Mr. Robert Shaw Green (Project Engineer - hardware related studies), Mrs. Renee Jasper, Mr. Marvin Katz, Dr. Jack Minker (Program Manager), Mr. James Mulford, and Mr. Warren E. Shindle (Project Engineer - problem analysis and programming related studies). The authors would like to express their appreciation to the numerous members of the AUERBACH Technical Staff who were called upon for expert advice on specialized areas of study. These are: Mr. George Neborak, Mr. Sheldon Einhorn, Mr. Don Russell, Mr. Ken Rose, Mr. Lawrence Feidelman, Mr. Ralph Howe, Mr. Arthur Hughes, Mr. Stephen Leibholz, Mr. Lester Probst, Mr. Robert Rossheim, and Dr. Jerome Sable.

Release of subject report to the general public is prohibited by the Strategic Trade Control Program, Mutual Defense Assistance Control List (revised 6 Jan. 65) published by the Department of State.

This report has been reviewed and is approved.

Approved: Howard Davis
ROBERT J. QUINN, JR.
Colonel, USAF
Chief, Intel and Info
Processing Div

Approved: [illegible]
FRANK J. TOMAINI
Chief, Info Processing Branch

FOR THE COMMANDER: [illegible]
IRVING J. GABELMAN
Chief, Advanced Studies Group

ABSTRACT

The objective of this effort was to determine the effect of associative memories which are realizable today to aid in processing formatted record problems. The evaluation consisted of a comparison between the CDC 1604B and the CDC 1604B-Associative Memory to process the same problem. The Goodyear Aerospace Corporation associative memory was used to establish state-of-the-art in associative memories, however, other associative memory designs were investigated.

PREVIOUS PAGE WAS BLANK, THEREFORE WAS NOT FILMED

TABLE OF CONTENTS
(Summary of Volumes I and II)

TABLE OF CONTENTS, VOLUME I

TABLE OF CONTENTS, VOLUME I (Cont.)

LIST OF ILLUSTRATIONS, VOLUME I

LIST OF TABLES, VOLUME I

EVALUATION

I. Study Objective:

The objective of this work was to determine the value of state-of-the-art associative memories to more efficient execution of typical non-numeric and intelligence problems by the general purpose computer. Thus it was not the intent to provide an absolute figure of merit which would set the value of small associative memories. Nor was it the intent to evaluate the full range of memory logic design or nature of connection to the computer which an associative memory device may take. It was hoped however, that a useful reference point would be established with which to guide further studies. It should be noted that these studies are the first significant attempts to evaluate an associative memory in a system context; * i.e., the associative memory is considered in association with a general purpose computer, typical peripheral hardware, and system programming.

II. Approach:

A. Selection of CDC 1604B and CDC 818 Disc

At the outset it was realized that the CDC 1604B computer, the CDC 818 disc and the Goodyear associative processor might not result in an optimum combination for the problems selected. Rather, the selection was based on the following:

1. The CDC 1604B is representative of "second generation" computers which are the most prominent in Air Force use today. Moreover, due to the high cost of replacing computer systems, it may be some time before present systems are replaced. Consequently, considerable savings might be realized from determining how these computers could be used more efficiently.

2. RADC recognized that there are several factors in addition to the choice of an associative memory and a general purpose computer which affect study results. These factors are associated with the software and with file structure, query statements, etc. thus by selecting the CDC 1604B and the CDC 818 disc, we provide the capability to modify the software and gain a more thorough understanding of the role of software in evaluating associative memories.

*See Mr. Ronald Ferris' comments in Volume II of the International Federation for Information Processing 1965 proceedings under the panel session on Content Addressable Memories.

3. Since the hardware is available at RADC, the software programs developed can be implemented and exercised in order to provide empirical data with a minimum of cost and time.

It should be emphasized that the selection of the CDC 1604B computer and CDC 818 disc does not affect the accuracy of the comparison made between the associative memory configuration and the non-associative memory configuration, since in both cases the same general purpose computer and disc were used. In fact, this is the type of data that is of interest to us.

B. Selection of Goodyear Associative Processor (GAP)

The primary intent of this effort was to evaluate the GAP merely as a first step in a long range plan to evaluate associative memories. Consequently, whether or not the GAP is optimally designed for a specific task is relatively unimportant. Instead it should be viewed as a research tool for use in obtaining valuable information pertaining to associative processing, particularly in the area of software. Accordingly the GAP design was selected with the intent to evaluate a state-of-the-art product with reasonable cost. Finally, it should be emphasized that evaluation of GAP cannot be construed as an evaluation of Goodyear Aerospace's capability to design computers.

C. Selection of Problems

The problems used throughout the study are actually processing functions. The intent was to choose a set of processing functions common to many specific problems and in so doing obtain generalizable results. Hence, the term sea surveillance merely indicates the source of the processing functions employed.

The selection of problems is a very important step and should not be taken lightly. The approach utilized in this study was briefly the following: *

1. Identify a class of problems within the Air Force

2. Identify the processing functions within that class and select the most common.

3. Based on a set of criteria; i.e., priority, availability of data base, etc. a final set of processing functions were selected. The criteria used in this analysis is as follows:

*This work was performed in-the-house at RADC. The details of this work are contained in a report located at RADC.

a. Does the product satisfy an operational requirement?

(1) Has a machine program been written to produce it?
(2) How many producers use it?
(3) Has the data base been built?

b. Are the program steps (operations, algorithm, etc.) common to other programs?

c. Is the program and data base description accessable?

(1) Is it releasable to contractor (Security) ?
(2) How soon can it be delivered to contractor (where do we get it and must it be sanitized) ?

d. Has program been written for 1604B class machine, i.e., is it one of these: 7090, 1410, 1218?

D. Evaluation Criteria and Measures

The following criteria and measures were developed by RADC, Auerbach Corporation and Goodyear Aerospace Corporation for use in their respective investigations:

Criteria	Measures
A. Complexity of Programming Required	1. No. of Instructions in program.
	2. Total time and cost of program.
	3. Time and cost of producing general flow charts (problem analysis & definition).
	4. Time and cost of detailed flow charts (required for time analysis).
B. Accuracy of Programming	1. No. of errors listed in (A3) and (A4) above.
	2. Type of errors located in (A3) and (A4) above.
C. Complexity of Hardware Required (AM)	1. Type and number of components used in AM e.g.; 100 transistors.
	2. Cost of AM and necessary interface.
	3. Time and cost for each AM macro instruction.
	4. Time and cost for each basic AM instruction.
	5. No. of AM searches required to complete each AM macro & basic instruction.

Criteria (Cont.)	Measures (Cont.)
D. Reliability of Hardware	1. Reliability of each component multiplied by some factors, e.g., no. of components.
E. System Efficiency	1. Time to execute program (machine run time).
	2. Total cost of programming, hardware, and machine running time.
	3. No. of AM & 1604B instructions used (and not used) and frequency of each.
	4. Amount of data that is read into and out of AM and frequency of transfers.
	5. Time AM is idle.
	6. Time required for 1604B to set up AM processing (data transfers etc.).
	7. Ratio of storage requirements vs availabilities.
	8. Time 1604B is idle.
	9. Time disc is idle. N/A
	10. % of relative running & idle times for the 1604B, Disc, & AM.

Key

N/A - Not Applicable to Goodyear

III. Use of Study Results:

The results of this study provide a good reference point when determining the usefulness of proposed associative memories. It is believed that this type of investigation, i.e., within a system context is a most important and necessary step. It is our hope that workers in the associative memory field can use this and future studies as a yardstick in designing associative memory - general purpose computer configurations. RADC encourages a close examination of criteria and measures to be used in evaluation of associative memories. We also encourage those interested to examine this effort critically and would make welcome suggestions.

[illegible]
RONALD J. FERRIS
Project Engineer

SECTION I. MANAGEMENT SUMMARY

1.1 INTRODUCTION

During the past few years a great deal of speculation has arisen concerning the effectiveness of an associative memory (AM), both large and small, within a computing system. The concept of an associative memory in which data is accessed by content rather than by its physical address has had considerable intuitive appeal for those interested in data manipulation rather than computational problems.

The technology associated with implementing such devices has made some progress. Several small associative memories have been implemented which contain approximately 2,048 words. Such memories have been constructed from so-called BILOC, BICORE, or similar elements. This technology is rather limited with respect to the core sizes that can be achieved at a reasonable cost. The technology associated with large memories is, on the other hand, being developed at the research level, using super-conductive elements.

Although considerable attention has been paid to the hardware aspects for both large and small associative memories, several other aspects of importance have not received adequate attention. Thus, if one reviews the extensive literature noted in the bibliography of this volume, very little may be found which describes the manner in which an AM can be integrated within a computer system, the programming tools that are required to take effective advantage of the memory, or the advantage or disadvantage of such memories in specific applications. Considering the early state of this growing technology, this is not surprising.

Rome Air Development Center (RADC) has developed a comprehensive plan to determine the advantages and disadvantages of both large and small associative memories. The work reported by the AUERBACH Corporation in these two volumes is part of this comprehensive plan. Specifically, the study has focused attention upon <u>small associative memories</u> as typified by the Goodyear Associative Memory (a description of the Goodyear Associative Processor (GAP) is contained in Appendix A of Volume II) within the context of a computing system consisting of a conventional computer (CDC 1604-B), discs (CDC 818), tapes, and conventional input-output equipment. Such a system has been termed by RADC a hybrid associative computer system.

As a vehicle for study, RADC has requested that the AUERBACH Corporation investigate a total data storage and retrieval system to determine the effectiveness of a so-called hybrid associative computer system for such a problem, and to answer some of the following questions:

(1) What is the most effective hybrid system for the problem studied?

(2) How can an associative memory be best integrated into the computer system?

(3) What are the costs associated with developing such devices?

(4) What possible variants of technology might enhance the AM or decrease the cost with small effect on the processing?

(5) How effective are the variant systems?

(6) Will any of the AMs substantially improve the overall system processing to warrant its use?

(7) Is a small associative memory effective?

(8) What is the influence of an AM on programming?

(9) What programming tools are required to facilitate use of an AM?

This management report provides the answers and limitations upon the answers to the above questions. References to Volume II, which contains detailed technical support, are provided so that the reader may delve further into technical material, if he so desires.

1.1.1 Scope of Study

This study is only one part of a larger and more comprehensive effort being directed by RADC, and while this investigation has a clearly defined and self-contained goal, the larger context must be considered. Insights and specific results and conclusions of this study are reported to RADC whether they are of primary importance to the fulfillment of the contract or a secondary result of the study. One of the goals of the study is to aid RADC in the evaluation of the applicability and usefulness of a class of small associative memories available in the present state of the technology.

Within the framework of the larger RADC effort, the overall goal of this investigation may be stated: Evaluate the applicability, effectiveness, and efficiency of hybrid associative processors. A sufficient range of hybrid organizations and depth of analysis is to be included to ensure a concrete and positive evaluation. The investigation is to consider both programming and equipment factors.

In addition to the overall goals specified above, several criteria for aiding in the evaluation of the AMs studied were specified by RADC; quantitative measures which should be obtained to aid in the evaluations were also specified. Such measures as number of instructions, number of transistors and flip flops, and time to perform algorithms for both hybrid associative configurations and conventional systems were utilized as quantitative measures and were developed. Some measures, such as cost for each basic AM instruction, cannot be answered. Once a basic AM capability exists, the cost of a particular AM instruction may be insignificant. How one apportions the cost of an instruction to the overall basic capability could not be determined adequately. The study was conducted using as a basis a problem termed sea surveillance and defined in the Office of Naval Research (ONR) document entitled, "Sea Surveillance Data Base Representation as Test Vehicle." The reasons for selecting this problem are given in the subsequent paragraphs. Within this context, decisions have been made affecting the specific scope, methodology, and approach of this and other AM research projects. Such decisions are noted in subsequent paragraphs of this section.

1.1.2 Problem Selection

The specific problem utilized as a study vehicle was selected by RADC. It was recognized that the study results would depend upon the potential that a hybrid associative computer system might have to aid the problem selected. The problem area selected for study was the storage of data and the retrieval of formatted files relevant to the intelligence community.

Before selecting a specific problem for study, RADC developed several criteria which the selected problem should satisfy. The problem should be an actual one within the intelligence community; a known conventional solution should exist; the system should be operational, should have known statistics concerning such factors as the number of files, size of data base, and queries posed by the user; and the data base should not be so small as to be trivially accommodated by current equipment and techniques. Several potential data bases were reviewed (see Paragraph 1.3 of Volume II), and a problem termed sea surveillance was selected for study. This problem satisfied most of the selection criteria, and, in addition, an unclassified version of this problem was available and defined in the aforementioned ONR document. Several solutions to the sea surveillance problem are known to exist. (Note: The sea surveillance problem reported in this document is related only to the above referenced ONR document and may bear no resemblance to any actual sea surveillance problem which may exist.)

1.1.3 Study Approach*

Several approaches can be taken in investigating the effectiveness of small associative memories. Each approach has certain advantages and disadvantages. One approach widely used to evaluate alternative conventional equipment is to select several complicated problems and to code these problems in detail for the alternative configurations. Advantages to such an approach are:

(1) A large number of manageable problems can be defined, detailed, and analyzed.

(2) Alternative situations can be considered, e.g., alternative data structures, record and file organizations.

*Also also evaluation

(3) Alternative problems bearing upon several subject areas can be considered, e. g., compiler developments, network flow problems, and data retrieval.

Disadvantages attendant with such an approach are:

(1) The importance that should be attached to each individual problem is difficult to define since its frequency of use in a particular system may be difficult to assess.

(2) Problems of importance to specific areas of interest on a particular problem may not be covered.

A second approach is to select a single comprehensive problem, potentially rich in associative memory operations, for study. This approach has several advantages:

(1) A comprehensive system is viewed for all data processing functions so that the frequency of operations is known and no processes are overlooked.

(2) Areas which may not have been selected for associative memory operations may be uncovered by reviewing all other processes.

(3) Assumptions as to the location of data which may tend to bias a solution towards an associative memory are avoided since one must take cognizance of the precise location of data as developed in previous processing steps.

(4) The associative memory can be utilized to store more than one class of data as required by dynamic system processing.

As with the aforementioned approach, there are certain inherent disadvantages. Some of these are:

(1) The solution reached for a particular problem may be too specific and may preclude generalization to other problems.

(2) The specific problem may be such that the full extent of the AM is not utilized.

(3) Alternative solutions might not be evaluated sufficiently since adequate time for investigating each alternative might not be available.

(4) Excessive time might be spent in systems design of the problem and, as a result, take time away from study of the details of associative memory processing.

Realizing the advantages and disadvantages attendant with each approach, RADC selected the latter for the AUERBACH Corporation and the former for other contractors. The study realized all of the advantages in this approach. By being forced to think at both a systems and a detailed level, many observations concerning associative memory processing have been developed. At the same time, several of the above disadvantages are pertinent. The solution reached is believed to be sufficiently general and applicable to problems of the same class. However, the queries and maintenance operations specified as 'typical' in the problem selected by RADC for study (the sea surveillance problem), did not sufficiently exercise the AMs studied. That is, the queries were very simple and did not require much processing while the data resided in the associative memory. More complex questions could have shown the AMs to greater advantage. Alternative solutions (e.g., different file structures) could not be studied. Thus, perhaps a more clever file structure than that utilized in this study could have been devised which would have shown the AMs to greater advantage. The study team cannot assess that since sufficient time did not exist for exploring this possibility. They can verify that some time was spent in systems design which could have been spent more profitably on the details of AM programming. In spite of this, much was learned in terms of organizing a system around an AM and programming for associative memories. The study team believes that, in the main, the advantages of the approach outweighed the disadvantages.

1.1.3.1 <u>Hardware Approach.</u> RADC has purchased an experimental associative memory from the Goodyear Aerospace Corporation. This AM has been termed the Goodyear Associative Processor (GAP) and is to be attached to the 1604-B at RADC. AUERBACH was asked to utilize the GAP in their study, and, more broadly, to investigate variations of the memory based upon the Goodyear technology and within the state of the art.

There are several AM variants that warrant study. These are:

(1) Logical Organization of the AM. The variations range from AMs which have minimal control units requiring detailed control by instruction from the 1604-B to ones which are independent of the 1604-B.

(2) Interface with the 1604-B. The variations possible here range from one which requires no modification to the 1604-B to one which has a special channel and special register and finally to one fully integrated with the 1604-B.

(3) Instruction Set. The possible variations range from a minimal set, which would decrease costs of the AM, to one which has extensive search capabilities and which generalizes the GAP capabilities.

The specific memories studied were developed after extensive meetings between programmers and the hardware designers. Cost analysis played a major role in the selection of the specific variants that were to be exercised in the study. To determine the costs involved, the GAP technology was studied thoroughly, logic diagrams were developed, and estimates made of the amount of equipment that would be required. AUERBACH's knowledge of the state of the art of integrated circuit technology and BILOC devices was utilized. Wherever the cost analysis was to show a minor increase in cost to implement a particular capability desired by a programmer, the feature was added. Cost analysis information is provided in Section II, Volume II.

To determine those associative memories that should be studied, a mechanization and configuration matrix depicting possible configurations was developed (Table 1-1). The range of hardware was to be from a basic minimum capability which can still operate as an AM through one which represents a full parallel capability. The interface with the central processor represented a second major variable to consider. By selecting systems represented by elements of the matrix for study, one could estimate the effect of these systems not covered, and the importance of the feature for the sea surveillance problem. Time prevented a study of all possibilities; a specific area which was not studied was the possibility of utilizing thin film for the response store. The memories specifically investigated are:

TABLE 1-1. AM LOGIC MECHANIZATION POSSIBILITIES

Connection and Configuration Possibilities		Minimum Hardware	Minimum Parallel Hardware	Full Parallel Hardware	Response Store in Thin Film
Peripheral Device	Buffer Channel	P	NP	NP	NP
	Transfer Channel	P (A1)	P (A2)	P	P
	Special IMA Channel	NP	P	P (GAP)	P
Multiprocessor		NP	NP	P*	P*
Integrated		NP	P	P (A3)	P*
Special I/O or Search Controlier		NP	NP	P*	P*

Key:

NP: Not a practical system (unbalanced).

P: A practical system worthy of study.

() : The notations in ( ) are the designations of the configuration studied in depth.

* : Indicates a configuration of especial promise and interest.

(1) A1 — an AUERBACH designed memory which represents a minimum hardware system interconnected with the CDC 1604-B via a direct transfer channel.

(2) A2 — an AUERBACH designed memory which represents a system with minimum parallel hardware interconnected with the CDC 1604-B via a direct transfer channel. The A2 memory contains eight tag bits while no other memory studied contains tag bits.

(3) GAP — a Goodyear Aerospace Corporation memory which represents a system with full parallel hardware interconnected with the CDC 1604-B via a special Direct Memory Access (DMA) channel.

(4) A3 — an AUERBACH designed memory which represents a system with full parallel hardware fully integrated with the CDC 1604-B. By fully integrated, it is meant that the last 2,048 words of CDC 1604-B core have been replaced by AM and can be utilized both as conventional core and as an associative memory.

The AUERBACH designed memories are feasible and are based on the technology developed by the Goodyear Aerospace Corporation. These memories are described in detail in Section II of Volume II, and a comparison of the configurations is given in Section II of this volume.

The basic elements of an associative memory are:

(1) The memory array — which provides the data storage itself.

(2) The comparand register — which contains the data to be compared against the contents of the memory array for searches, provides a shifting register for some operations, and can play an intermediate role in the transfer of data between the memory array and the central processor.

(3) The mask register — which is used to contain data specifying portions of words for operations involving only word portions.

(4) The resolver — which is used to determine the location of response bits in the response store.

(5) The search logic — which causes the search commands received by the memory to be properly executed.

(6) The response store — which receives vectors indicating which data satisfy a given search criterion, and which can execute logical operations, such as shifting and Boolean operations, on these vectors.

In addition, an associative control capability is provided. This capability is utilized to decode instructions.

These elements are presented in Figure 2-1.

The search logic and response vector's storage registers in the Goodyear Associative Processor count for approximately 45 percent of the production cost of the GAP. It is clear that large reductions can be made in the amount of hardware associated with the response store. Since it was one of the objectives of this study to evaluate both sophisticated and unsophisticated associative memory features, some features were expanded and others eliminated in the configurations studied.

1.1.3.2 Software Approach. The sea surveillance problem referenced in the ONR document lists a small set of queries that must be answered (see Paragraph 3.5, Section III, Volume II). One can organize a system which can handle the specific queries listed in an efficient manner while other queries are handled inefficiently. Such an approach could not be taken for this study since the results obtained would be useless for problems of a similar nature to sea surveillance. Indeed, the nature of intelligence problems is such that if a system were based exclusively on the recognized requirements, and not generalized, other requirements would shortly arise possibly obviating what was achieved and requiring extensive reprogramming.

The approach taken was to design a system that was not biased towards any specific queries, but organized to handle a wide variety of queries that the user could specify. Thus, the user is permitted to specify queries in which the search criterion is any logical condition which contains the logical connectives AND, OR, and NOT to any complexity he desires. Within this context, several of the queries specified in Paragraph 3.5 of Volume II have been timed both for AM configurations and non-AM configurations. Queries other than those specified (e.g., having arbitrary logical conditions) can also be effected utilizing the mechanism developed in this study. In addition to queries, consideration has been given to the problem of new inputs and maintenance. Maintenance orders and new inputs can be timed, if so desired.

To develop a unified approach for the sea surveillance problem to be applicable for all configurations was a challenge. Although a unified approach was achieved, all features of each AM could not be studied in detail. However, GAP was studied fully. A generalized file structure has been developed which is applicable for all configurations. The file structure is biased towards AM processing. In addition, an executive control routine which analyzes queries and handles them in a general manner was developed as was a general approach for input processing and maintenance. To show the variation in AM and non-AM configurations, machine code was developed for GAP and non-AM configurations. Coding for the variant AMs was not performed explicitly In one instance, the instruction complement of the AM (A3) was such that GAP coding was a subset of the AM and its effect could be estimated; in a second instance, the GAP was "simulated" and the variation in timing noted.

1.1.4 Study Restrictions

Several study restrictions are important to note as they affect the generality of the results. The restrictions imposed by RADC are as follows:

(1) Small Associative Memories. The study was restricted to include only associative memories within the state of the art. Thus, only small associative memories have been investigated. In particular, the Goodyear Associative Processor was specified by RADC as typical of the technology of associative memories. The observations and conclusions developed, therefore, do not necessarily extend to large associative memories. The hardware techniques for large AMs will be considerably different than those of GAP.

(2) Central Processor. The central processor to be utilized as typical was the CDC 1604-B. It should not be construed that this is an ideal central processor. No attempt was made to determine an "ideal" central

processor. Features of the 1604-B that proved cumbersome are noted.

(5) Disc Unit. RADC selected the CDC 818 Disc for the study since it is attached to the 1604-B at the RADC facility. The file structure developed to focus-in upon the data was tailored towards the peculiarities of this disc to avoid bias. For a different disc, a modified file structure would have to be developed.

1.2 FINDINGS

This paragraph presents the study findings in summary form, with brief explanations. (Volume II provides ample supplementary information to support the findings.) Where pertinent, references to Volume II are provided. The study findings fall into three broad areas of interest:

(1) Hardware — findings with respect to the nature, organization, utility, and value of features of associative memories.

(2) Programming — findings with respect to using an associative hybrid processor for the performance of defined tasks and subtasks.

(3) Concepts of Systems — findings affecting the concepts underlying associative processing, the design of data bases, and general effectiveness of AM processing.

In general, the findings pertain only to the particular class of problem studied and reported upon in these volumes. Extension of the comments to other applications may be warranted, but requires further study. The more important findings are noted in this paragraph by capital letters and underscored.

1.2.1 Hardware Findings

Desirable and non-desirable hardware findings were obtained as a result of designing and programming the sea surveillance system. One of the most important of these findings concerns the interface provided for GAP.

THE EFFECTIVENESS OF AN AM FOR PROBLEMS SUCH AS SEA SURVEILLANCE IS HIGHLY DEPENDENT UPON THE AM INTERFACE WITH THE DIGITAL COMPUTER. THE A3 WAS FOUND TO HAVE THE BEST INTERFACE.

The sea surveillance problem is primarily input-output limited. Because data transfers to AM go through core, GAP was found more time consuming for performing the simple queries studied than a non-AM solution; the A3 configuration, which was fully integrated with the 1604-B, required the same amount of time. In the A3 configuration the data base is stored on disc. To take full advantage of the AM requires direct transfer of data from disc to AM without intermediate core transfers. More complex queries would show A3 to advantage over the non-AM solution and GAP. The A3 configuration was found to have the best interface. In the A3 configuration, the associative memory is fully integrated into the 1604 memory as shown in Figure 1-1. In GAP, the associative memory is attached to the direct mem[illegible]y access channel, much as one would attach another unit to the computer control. In A[illegible] and A2, the memories are attached to input-output Channel 7, much as peripheral devices are attached.

The interface between GAP and the 1604 is via a direct memory access channel (DMA).

THE USE OF THE DIRECT MEMORY ACCESS CHANNEL AND THE "MULTIPROCESSING" CAPABILITY PROVIDED FOR THE GAP HYBRID CONFIGURATION HAD MAJOR SHORTCOMINGS.

This finding was contrary to some expectations and is based upon the following:

(1) The overhead needed to set[illegible], control, coordinate, and synchronize GAP, because it operates as a "multiprocessor" on the DMA, is very high (Paragraph 2.5.5.1, Section II, Volume II).

(2) In some cases, contrary to programmer intuition, it is better to code a problem using inelegant coding rather than exiting from GAP coding to perform 1604-B coding and returning to GAP. This arises because of the overhead associated with the DMA "multiprocessing" character of GAP. An illustration of such coding is shown in Paragraphs 7.2.1.2 and 7.3.5 of Volume II.

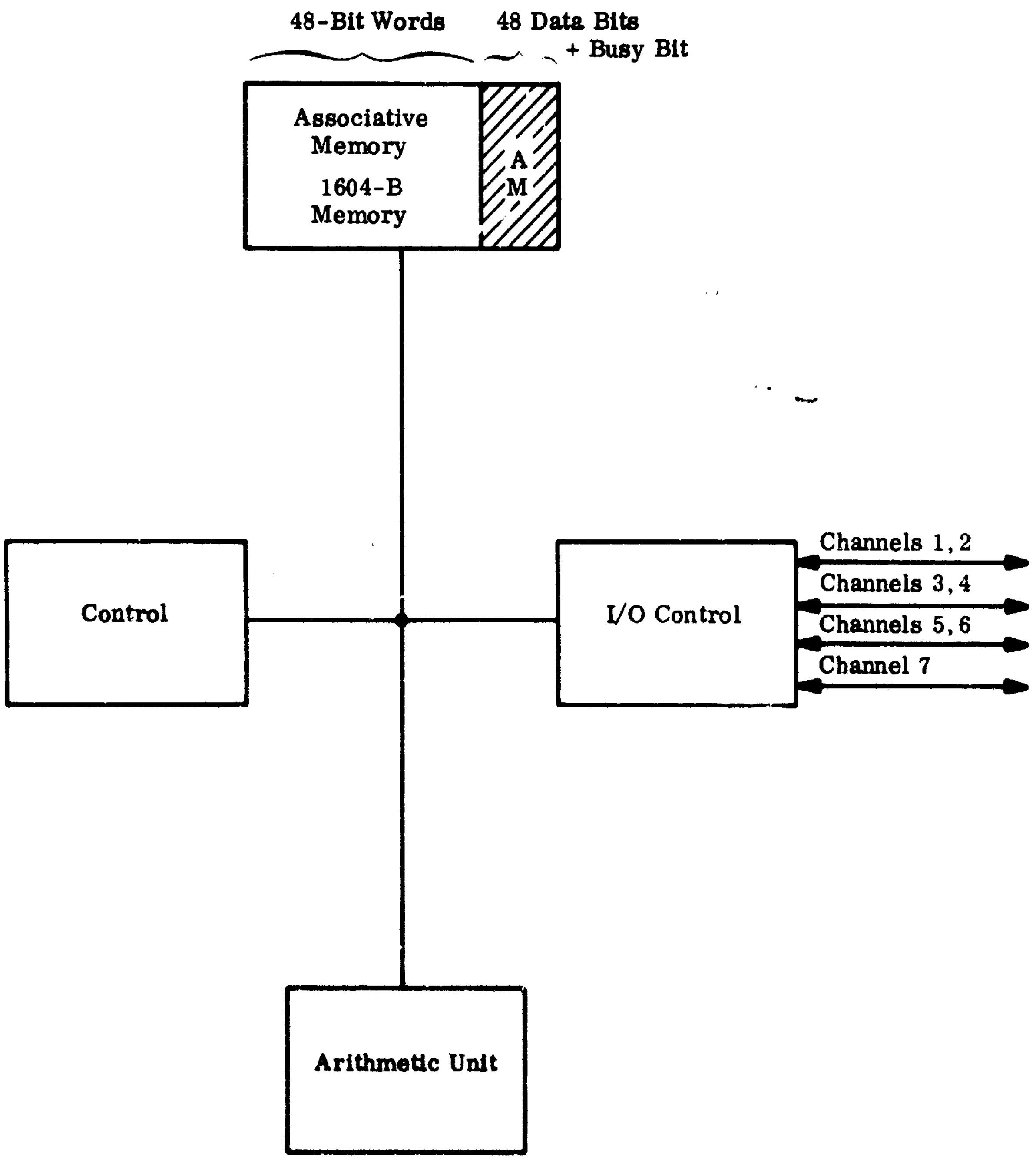


Figure 1-1. Place of Associative Memory in Internal A3 Configuration of 1604-B

(3) Lack of an interrupt capability limits communication between GAP and the 1604-B and seriously limits GAP as a "multiprocessor" on the DMA channel. An interrupt on Halt would improve GAP-1604-B communications.

The cost of an associative processor was an item of major importance to the study. The most important finding with respect to cost follows.

THE ELEMENTS OF COST OF AN ASSOCIATIVE MEMORY ARE THE ASSOCIATIVE MEMORY ARRAY AND ASSOCIATED ELECTRONICS; RESPONSE STORE; AND REGISTERS, GATES, AND CONTROLS. THE RESPONSE STORE CAN BE VARIED SIGNIFICANTLY TO DECREASE COSTS.

A cost analysis was performed on each configuration studied. The relative costs are shown in Figure 1-2. The direct interface of A3 with the 1604-B and increased instruction capability surprisingly do not increase the cost of the AM. Since all AMs studied utilize the Goodyear technology, the cost of the basic array is the same. Note that one can afford approximately twice as much AM storage capacity in A2 for the price of one GAP.

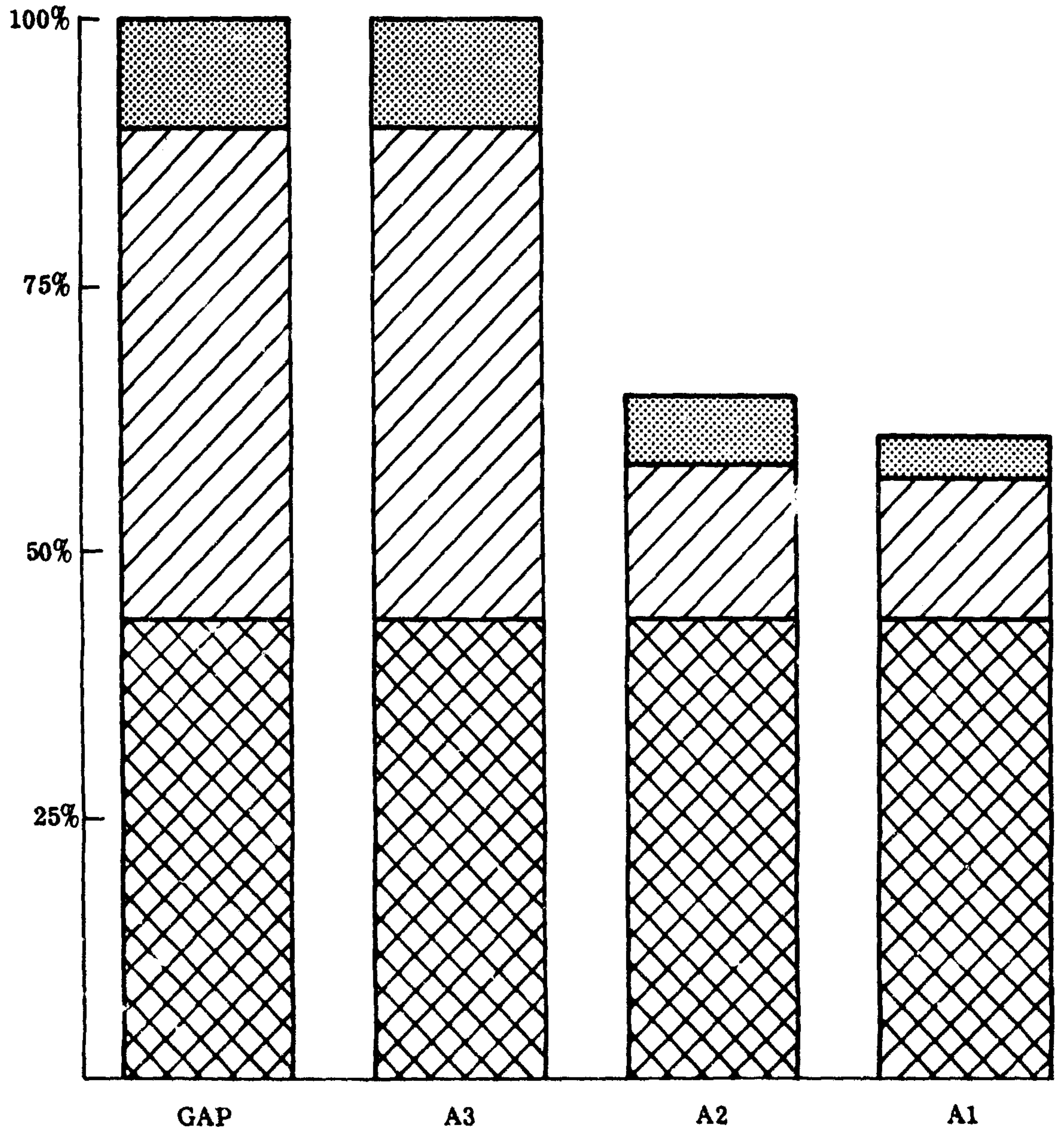
The capability to communicate results of an operation to the program coding itself is the basis of data dependent processing. This capability can be called the intra-communication capability of a processor.

GAP LACKS INTRACOMMUNICATION CAPABILITIES.

AM-to-AM transfers, plus AM modification by index registers, proved a limitation in the data manipulation. The A3 configuration is not so restricted nor is A2 (but A2 is restricted to a lesser degree than A3). Data-dependent processing as required by sea surveillance type problems requires this feature (see Paragraph 7.3.5, Volume II).

Some hardware features of the GAP, particularly the "buffers", proved valuable.

THE "D" AND "E" BUFFERS IN GAP PROVIDE SIGNIFICANT CAPABILITIES AND CAN BE GENERALIZED.



LEGEND:

▩	Memory Array and Associated Electronics
▨	Response Store
▒	Registers, Gates and Control

Figure 1-2. Relative Costs: GAP, A3, A2, and A1

The P, T, V, and Z fields of GAP instructions permit the programmer to take advantage of the capability. Capabilities in the D and E buffers are generalized in A3 over GAP. The following also pertains:

(1) This capability proved less useful than the capability of storing all response vectors in 1604 memory, or in tag bits, until needed. This observation may, however, be the result of the peculiarities of the data base design studied.

(2) The capability to shift down D and E buffers and response store is essential. The capability to shift buffers up is desirable, but may not be worth the cost.

Hardware provided for A2 only proved worthwhile, especially in the built-in features for using the tag memory concept. The following finding was supported by the study of A2.

TAG BITS ARE IMPORTANT AND ARE REQUIRED.

The absence of tag bits proved a major limitation in GAP and A3, although substitutes could be found in both cases. By utilizing the second half of GAP as tag bits, one achieves this result. Actual tag bits generalized as in A2 are a less costly solution than wasting the entire second half of GAP for a tag memory. On the other hand, some deficiencies in the instruction repertoire of GAP were discovered.

DESIGN OF GAP INSTRUCTIONS REQUIRES IMPROVEMENT.

(1) Manipulation of the LDR instruction was found to be cumbersome. This instruction is utilized so frequently as to require improvement.

(2) The relatively complicated searches, ~~such as NHC (next~~ higher than comparand) and ALC (next lower than comparand) proved to be easy to program and might not be worth additional cost.

(3) Readout of addresses and count of responders should be inserted into the desired 1604-B word rather than fill the unused bits of the words with zeros.

1.2.2 Programming Findings

The study of the AM hybrid configurations in solving the sea surveillance problem gave ample opportunity for collecting data about programming. This paragraph reviews the most important findings about programming uncovered by the study. The findings are relevant to programmer learning experience, ease of programming, most used and least used features, most common programming problems, software tools required, and efficiency of AM hybrid coding. No results could be obtained on the accuracy of coding since no actual machine testing was possible.

1.2.2.1 Programmer Learning Experience. Programmers assigned to the sea surveillance problem were experienced senior program designers familiar with a wide variety of conventional and unconventional processors. ALL PROGRAMMERS FOUND GAP DIFFICULT TO LEARN. There appear to be two major causes of learning difficulty. First, the GAP programming manual (Appendix A, Volume II) is deficient. It lacks a detailed description of the effect of instructions on registers and the store, and is poorly organized from both a pedagogical and reference point of view.* Second, the associative configurations are conceptually different from other configurations. All associative configurations studied could be used in a wide variety of ways, such as tag memories, as "look-up" machines, as logic machines, and, to a surprising extent, as conventional serial processors. Hence, the programmer was confronted with a wide variety of concepts in a new context. Learning to use these concepts involved considerable trial and error, and some of the most powerful programming tools were mastered relatively late in the project.

Although the programming staff had full access to the developers of the A1, A2, and A3 memories, it is also true that these memories proved difficult to learn. THE MAJOR CAUSE OF LEARNING DIFFICULTY IS THEREFORE THE STRANGENESS OF ASSOCIATIVE PROCESSING TO PROGRAMMERS. All programmers engaged in the project felt that, with more experience, they could probably meaningfully improve the quality of their output.

1.2.2.2 Ease of Programming. Although each programmer has a feeling for the relative difficulty of programming a machine with which he is familiar, there is no one measure of the relative ease or difficulty of programming a particular machine.

*This was to be expected because of time constraints. The programmer manual received was a draft copy.

Some of the important factors in determining ease of programming include the number of instructions and storage locations affected by an instruction, the frequency of iterative loops required for a task, the relative complexity of flowcharts, and the number of fields that must be specified for an instruction. There is no convenient way to combine these measures into one measure for relative ease of programming. Programmers found, however, that certain tasks were easier to program for an associative processor than for a conventional configuration. A few tasks proved more difficult to program for an associative processor.

PROGRAMMING FILE MAINTENANCE OPERATIONS PROVED EASIER, BECAUSE SEQUENCING OF DATA FIELDS WAS USUALLY UNNECESSARY. Because the associative memory makes addressing data by its contents convenient, the logically involved operations required to locate a data field in the proper place within a record were greatly simplified. The data field value was placed ajacent to the field definition, and because of the ability of an AM to search the entire record at one time, it became unnecessary to locate the field in some particular place within the record. Since order could be ignored in these cases, the programming required to maintain order could be bypassed.

Some tasks were proven not suitable for associative processing. These tasks include tasks requiring arithmetic processing such as calculating distance from geodetic position data, and data-dependent processing. FOR ALL TASKS EXCEPT ARITHMETIC AND DATA DEPENDENT PROCESSING, FEWER INSTRUCTIONS NEEDED TO BE WRITTEN BY THE PROGRAMMER THAN WERE REQUIRED FOR CONVENTIONAL PROCESSING. This fact is derived from some peculiarities of associative processing. FIRST, FEWER LOOPS ARE REQUIRED, BECAUSE OPERATIONS CAN BE PERFORMED IN PARALLEL, AND HENCE THE CODING NEEDED TO SET UP AND CONTROL THE LOOPS NEED NOT BE WRITTEN. SECOND, ONE CAN PERFORM AN OPERATION ON SEVERAL DATA FIELDS AND EVEN SEVERAL DATA RECORDS AT THE SAME TIME. Hence, the coding usually required to step through fields and records one at a time can be omitted.

The difference in the number of instructions required to program representative query macros is shown in Figure 1-3. In this figure the number of instructions written for GAP is compared to the number of instructions written for the same macros for the 1604-B. The reader must be very careful about generalizing the data of this

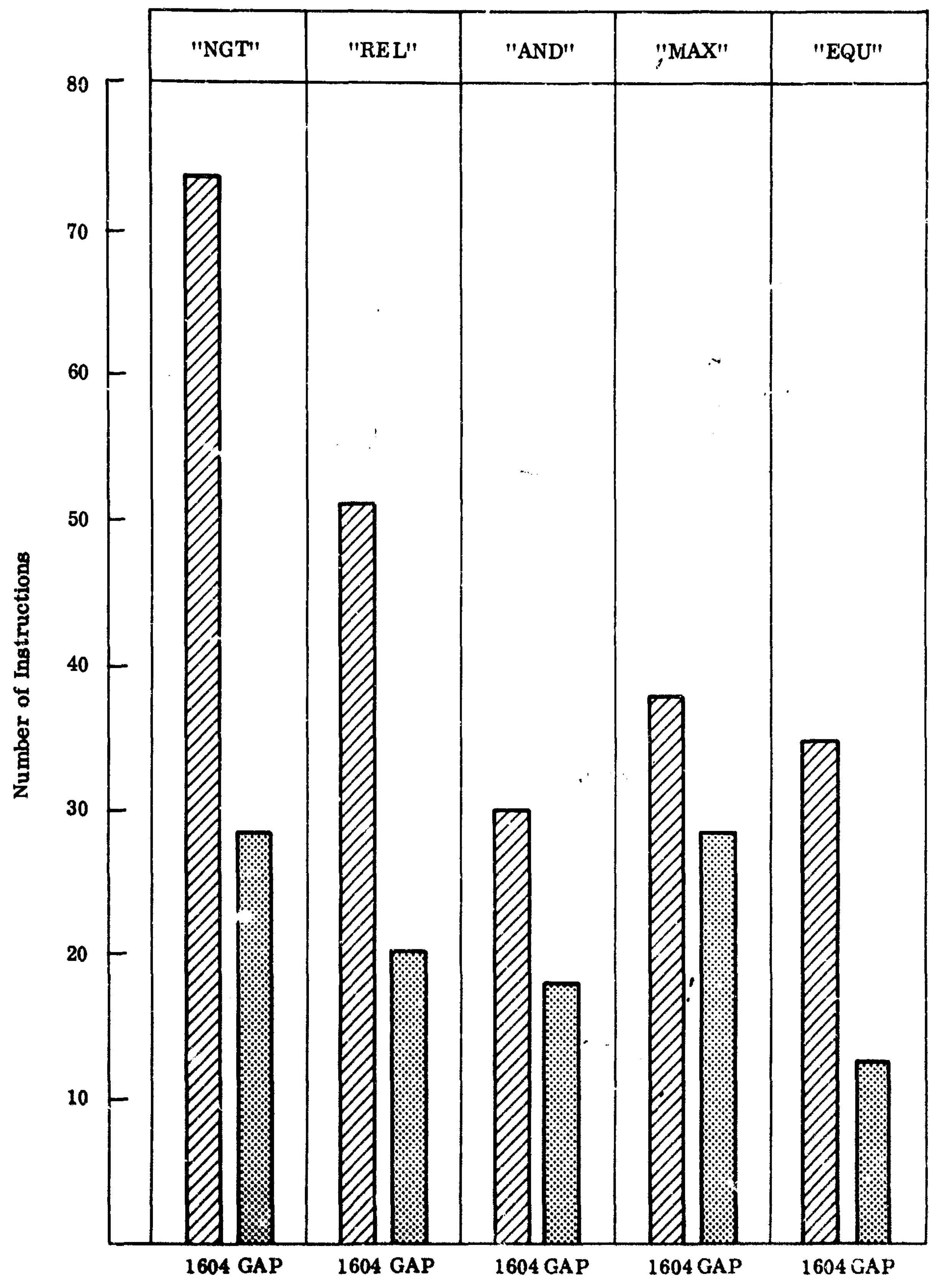


Figure 1-3. Number of Programming Instructions Coded for Query Macros, GAP vs. 1604

figure. The records processed by the macros were designed in light of the AM, and, specifically, order constraints which do not exist for the AM were ignored in specifying the records to be processed. As a result, 1604 coding to perform the various macros includes numerous housekeeping steps not required when the data records are designed for conventional processors.* These housekeeping operations somewhat inflate the figures for the number of instructions required for 1604 processing.

The relative infrequency of loops and the capability to process several fields and records simultaneously resulted in some differences of flowcharting. FLOWCHARTS REQUIRED FEWER FIXED CONNECTORS, AND THEREFORE APPEARED SIMPLER. ON THE OTHER HAND, NO NEW FLOWCHARTING CONVENTIONS WERE NEEDED. Except for the use of a few new phrases, such as "set responders for ...." or "AND response store with the .... buffer," the usual process boxes and decision boxes sufficed.

The number of fields required to specify an instruction to GAP and to A3 seemed excessive, although this is perhaps a result of the strangeness of the instructions. All programmers found it inconvenient to express S, N, and R in an LDR instruction for GAP before every load. Further, the use of the letters "P," "T," "V," and "Z" for fields in GAP is certainly not mnemonic, and may contribute to the feeling that the GAP instructions are "too complex."

Programmers found it difficult to determine the quality of their coding, since the relative efficiency of an algorithm was often contrary to intuition. There are several causes of this condition. First, an apparently inefficient algorithm can often minimize transfers of control between the GAP and 1604. WHENEVER THE TRANSFER OF CONTROL BETWEEN GAP AND THE 1604 IS MINIMIZED, THE EFFICIENCY IS IMPROVED. Consequently, it is often best to do a certain amount of processing inefficiently in GAP rather than suffer the penalty attendant on transferring control. Second, TRANSFERS OF DATA BETWEEN THE GAP AND THE 1604 ARE QUITE COSTLY IN TERMS OF TIME, and savings in time gained by the parallelism of the GAP can be made up for several fold by the need for data transfers.

*The results of this report are deficient in this regard, i.e., it was the expressed desire of RADC to optimally design the conventional system.

THE MOST SERIOUS DEFICIENCY OF THE GAP INSTRUCTION REPERTOIRE, FROM THE POINT OF VIEW OF THE PROGRAMMER, IS THE ABSENCE OF A FULL CAPABILITY FOR HANDLING DATA DEPENDENT OPERATIONS. It is usually necessary to use the 1604 to set up a coding sequence for GAP, owing, in many cases, to relatively minor faults in the conception of GAP. The Read Address of Responders instruction zero fills the word into which the address is read, making the use of dummy instructions difficult. IN FACT, IT IS SO COSTLY TO ASSEMBLE INSTRUCTIONS USING GAP CODE THAT THE PROGRAMMER IS MOST OFTEN FORCED TO USE THE 1604 FOR THIS PURPOSE.

In processing data, the operations effected are often determined by the data itself. In conventional processing, the procedure often used is to assign a counter to each class of data and to test that datum corresponding to the particular counter's value. The counter is then advanced. In this scheme, it is possible to permit a datum to belong to more than one class Now in the GAP, the test for status of a particular class of data determines the status of all data in the class for each searchable (busy bit set) word. This mass of information often proved to be useless since the status of the "first" responder was of interest. Then after examining the first responder (and ignoring all others until later), it was erased. Consequently, the datum could only belong to one class. Additionally, any subsequent test destroyed the status of the first class of data. Consequently, a reset cycle had to be performed. CLEARLY, THE GOODYEAR ASSOCIATIVE PROCESSOR OFFERS NO ADVANTAGE IN EFFECTING DATA DEPENDENT PROCESSING.

1.3.2.3 Most Used and Least Used Features. THE MOST USED INSTRUCTIONS PROVED TO BE THE SIMPLE SEARCHES AND THE LOAD AND READ INSTRUCTIONS. THE LEAST USED INSTRUCTIONS WERE THE MORE COMPLEX SEARCHES. The simple searches almost always sufficed for establishing response vectors used in processing, and the more complex searches, such as next higher/lower than comparand, were not only seldom used, but could be easily programmed in any case. The complex search, "CPX," was not used in the study. CPX will prove valuable when many fields are contained in one word. In this system only one major field occurs in a word in most cases. THE LDR INSTRUCTION WAS OFTEN AN INCONVENIENCE, because it is required for the specification of fields about which the programmer usually does not care. For the vast majority of cases, S = 0 in the LDR, and R and N need not be specified, because the operation is to range over a half memory or the entire memory.

THE NEXT HIGHER/LOWER THAN COMPARAND, MAX, MIN, AND OTHER COMPLEX SEARCHES WERE LUXURIES. These searches could be programmed quite easily, and were seldom used in the problem at hand.

1.2.2.4 Most Common Programming Problems and Solutions. Certain problems were encountered so frequently in dealing with sea surveillance problems that standardized solutions were developed for them. Two of the problems fell into a class which required solutions for this specific problem, while a third class of problems seems to cast light on features desirable in compilers for associative processors.

A recurring problem for programmers using the hybrid configurations derives from the relative size of the AM. In those cases in which the data to be processed exceeds the storage of the AM, the programmer must repeat operations on different segments of the data. The problem is not trivial, since the number of times the AM must be loaded and the order in which operations are to be performed are variable. For example, if it would require only two memory loads to enter a list into AM, but the programmer requires that a Max search be performed on the list as if it were one memory load, then three separate memory loads will be required to perform the simulation of a larger memory. If, however, a search on equality (EMC) is required, only two loads will be required. The problem was solved by developing a routine that functions in the manner of certain executive routines; the routine is called the AM Dispatcher. It receives commands from a program using the AM, and successively performs operations for the program as if a longer AM were available. AN EXECUTIVE FUNCTIONAL ROUTINE WHICH WILL OVERLAY AM STORAGE FOR A PROGRAM WILL NO DOUBT PROVE DESIRABLE FOR MANY OTHER PROBLEMS, AS IT HAS FOR SEA SURVEILLANCE.

A second class of problems is derived from the peculiarities of GAP. Unless one is using GAP as a Tag Memory, it is surprisingly difficult to subset the data in the AM in order to perform a sequence of operations on that subset alone. This class of problems can be readily solved with a non-trivial macro, SPO. MACROS LIKE SPO, WHICH SERVE THE PURPOSE OF TEMPORARILY SUBSETTING THE DATA IN THE AM, WILL UNDOUBTEDLY PROVE OF VALUE IN MANY GAP SYSTEMS AS SPO HAS IN THIS ONE.

Finally, it is worth stating that the study showed some operations to be so frequent and of such general utility, as to perhaps influence the specification of compilers for hybrid configurations.

The functions most used are listed below:

(1) Identify all x (i. e., turn on responders for all entries identical to some masked portion of a field).

(2) Fetch from AM all those data satisfying a search criterion.

(3) Erase all entries satisfying criteria x.

(4) Substitute x for y in all responders.

1.2.2.5 Program Efficiency. To say that program A is more efficient than program B in performing a certain task is to say that program A will perform that task more economically. When input and output specifications are the same for both programs, and when the computer systems are of equal cost for program A and program B, efficiency is a function of processing time. As is well-known, some problems on some computer systems can be evaluated for efficiency as a simple function of the time required to input and/or output the data required. Such problem configuration systems are said to be "input-output bound."

When the sea surveillance problem is processed with the proposed data base stored on the 818 Disc, THE SYSTEM IS INPUT-OUTPUT BOUND, WHETHER AN ASSOCIATIVE MEMORY IS AVAILABLE OR NOT. The time to perform a task upon the specified system is so nearly a function of the time required to input and output disc data that the relative efficiency of the AM and non-AM configuration will be very nearly the same. It is therefore necessary to define efficiency independent of the 818 Disc. Such a definition is not easily obtained. The present system was designed to optimize processing of data stored on the 818 Disc. To pretend to ignore the disc system's influence upon the actual system efficiency is therefore misleading.

Keeping the influence of the disc in mind, it is still possible to get information about relative efficiency of an AM augmented system in contrast to one without the AM. It is convenient to begin an inquiry about efficiency by noting that it is possible for an AM itself to be penalized by input-output.

It is perfectly proper to speak of an associative memory as input-output bound if the processing time is very nearly a function of the time required to load and/or unload the associative memory. This is true whether the device is a proper peripheral device (A2), a device on the DMA channel (GAP), or a device integrated into memory but still perhaps requiring memory-to-memory transfers (A3).

The sea surveillance problem is such that, with the exception of a few functions in file maintenance, the time required to load and/or unload the AM is a very good indicator of the processing involved. FOR THE SEA SURVEILLANCE PROBLEM, WITH THE EXCEPTION OF FILE MAINTENANCE AND QUERY PROCESSING OPERATIONS, PROCESSING TIME IS NEARLY A FUNCTION OF LOADING TIME FOR GAP AND A2.

Ironically, the situation for GAP and A2 is worse than being input-output bound for this problem. LOADING AND UNLOADING GAP AND A2 CANNOT BE OVERLAPPED WITH PROCESSING. HENCE, GAP AND A2 ARE SEVERELY PENALIZED BY LOADING, RATHER THAN BEING BOUND BY IT. Processing time, which is relatively small, must be added to load and unload time.

Figure 1-4 graphically shows the effect of load time for GAP. The solid line shows the total amount of processing time per iteration without using an AM and assuming the data is in core; the broken line shows the total processing time, per iteration, for GAP, assuming that the data remains in AM for each successive iteration. The routine in question is "REL" and it is perfectly reasonable to assume that the routine would be repeatedly used to apply successive relational tests to elements in a list.

THE FIGURE SHOWS CONCLUSIVELY THAT PROCESSING TIME PER ITERATION DECREASES IF THE DATA REMAINS IN AM. This fact is a simple consequence of prorating the load/unload penalty over iterations. Further, it should be obvious that this fact can be generalized, since it should always be correct to prorate the load/unload penalty over successive iterations.

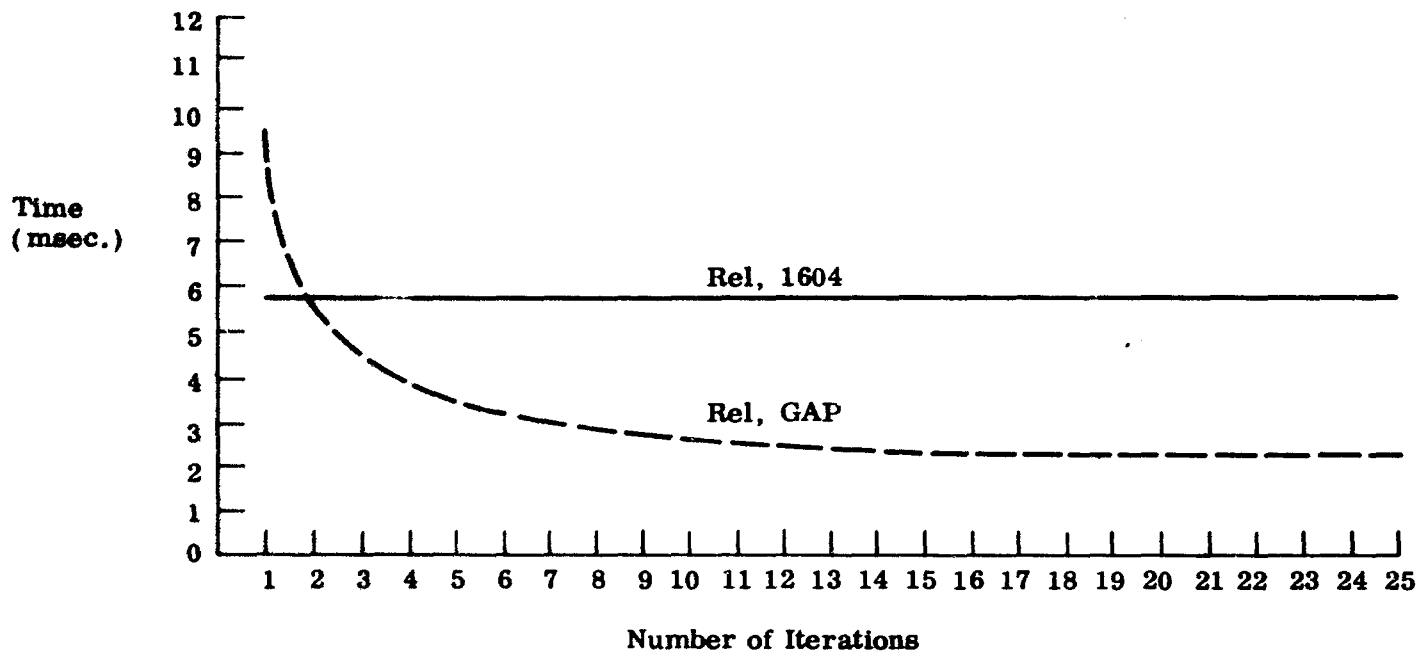


Figure 1-4. Load Time Prorated over Iterations

In Figure 1-4 the dotted and solid lines cross. This point can be called the iteration breakpoint. For almost all query functions over 1,000 data words, an iteration breakpoint exists. At the iteration breakpoint for the given data volume, it is more efficient to use an AM for query processing than to use the conventional processor. It should be noted, however, that only relatively complex queries will require enough iterations over the data to show the AM to advantage.

The above conclusion is further corroborated by Figure 1-5, which shows the effect of prorating the load/unload penalty over successive iterations for three other query processing routines.

IN THE SEA SURVEILLANCE SYSTEM, THE QUERIES WERE RELATIVELY SIMPLE. HENCE, THE GAP AND A2 MEMORIES ARE LESS EFFICIENT THAN A CONVENTIONAL MEMORY SYSTEM. ON THE OTHER HAND, IT IS EVIDENT THAT QUERIES CAN BE FORMULATED WHICH ARE BETTER PERFORMED IN THE AM THAN CONVENTIONALLY.

This fact is a simple consequence of the existence of iteration breakpoints, and the legitimacy of prorating the load/unload penalty over successively iterated AM operations.

The entire problem of the load/unload penalty arises because of the way in which A2 and GAP are configured. For both A2 and GAP, data input to the system enters the AM after a brief pause in the 1604 core. The routing of data through the 1604 is a result of the ability of either the DMA channel or Channel 7 to communicate directly with an input-output channel. Because the data must be routed through the 1604 and then loaded into the FM, the load/unload penalty exists.

Configuration A3 does not require that data be routed through the 1604. Data can be read directly into the A3 from the peripheral device, with only a small penalty due to the difference of the cycle times between core and the AM memory. A3 WILL PROVE TO BE ADVANTAGEOUS FOR QUERY PROCESSING WITH LESS COMPLEX QUERIES THAN REQUIRED FOR A2 AND GAP. HENCE, THE A3 CONFIGURATION IS "MORE EFFICIENT" THAN EITHER GAP OR A2 FOR PROBLEMS OF THIS KIND.

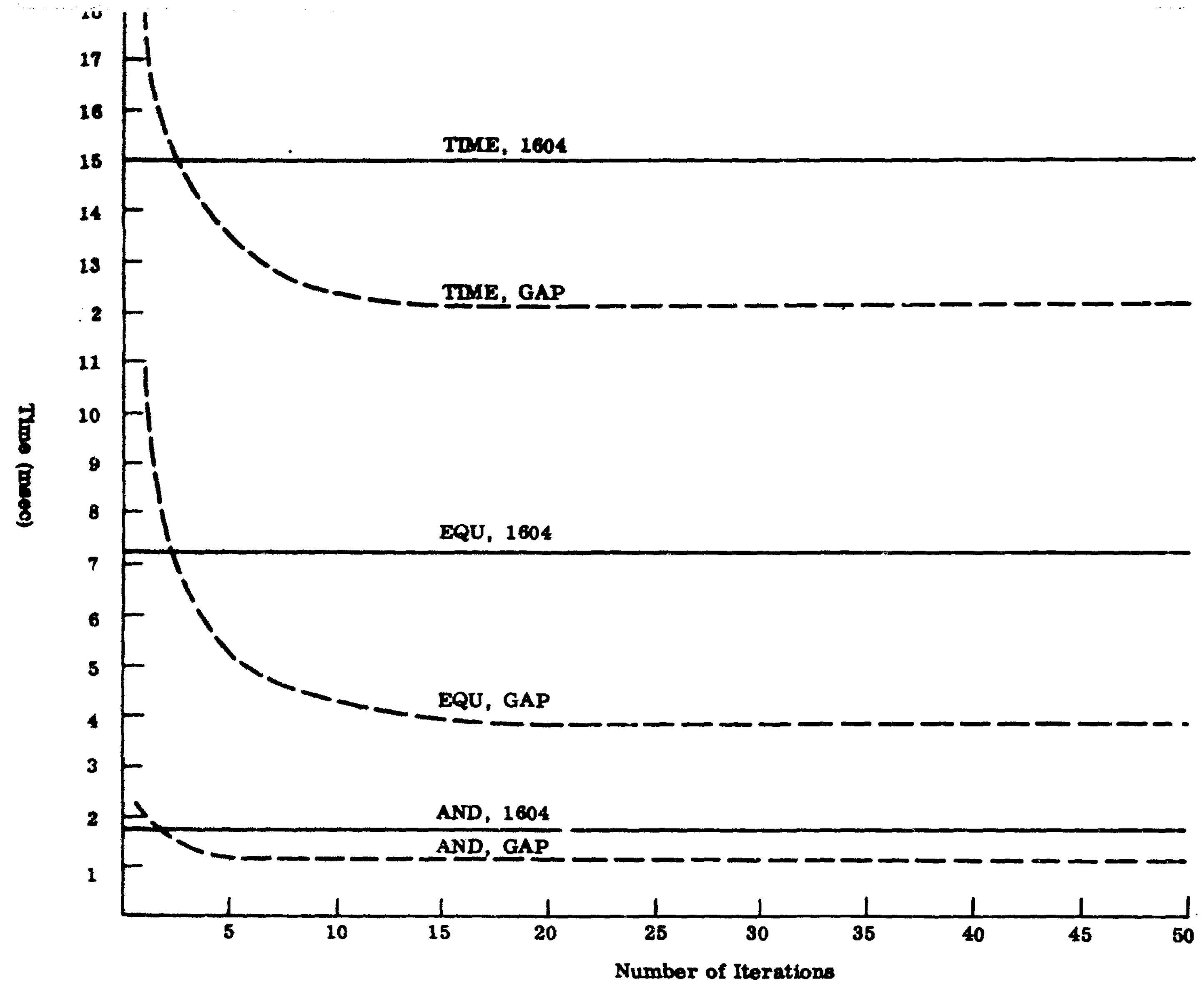


Figure 1-5. Iteration Graphs for Three Query Subroutines, TIME, EQU, and AND

The time required to execute the simple questions of sea surveillance, through query procesing alone, is shown in Figure 1-6. Queries of sufficient complexity to show GAP and A3 to advantage were not invented or timed as part of the study.

It should be noted, however, that the full realization of the A3 potential for minimizing the load/unload penalty of GAP and A2 requires the development of a relatively sophisticated input-output routine, in order to transfer data directly between the disc and AM.

GAP, as distinct from both A2 and A3, suffers one other penalty besides the load/unload. This penalty may be called the transfer penalty, which is paid in the microseconds required each time GAP is to begin processing a particular sequence of GAP coding and in the microseconds required to test the condition of the AM when a coding sequence is complete. Compared to the load/unload penalty, this penalty is minor, and is largely alleviated by a minor change recommended for GAP, the interrupt on halt.

1.2.3 Concept Findings

The availability of an associative memory changes the thinking of systems designers, and suggests alternative solutions to problems. The ability of such a device to stimulate the exploration of different ideas is perhaps its most important contribution to the field. Unfortunately, only a few of the various ideas suggested by the memory could be explored in a study of this kind. Avenues of further research thought to be of promise are presented in Paragraph 1.3 of this volume. This paragraph lis's some of the conceptual findings actually used in the course of the study.

The order and location of data within records are usually two of the important keys to processing and understanding the meaning of each item of data. The associative memory has the capability to process data without regard to the relative location of a datum within a record, by referencing the datum in terms of its contents or, in the case of the data base actually used, in terms of the name of the datum. THE DESIGNER OF THE DATA RECORD WAS FREED FROM CONSIDERATIONS OF ORDER AND RELATIVE LOCATION WITHIN RECORDS.

QUESTION	GAP	1604-B	A3 (best case only)
How many ships are within r miles of point p ?	361.3	357.9	358.7
Are any U. S. submarines in area _____________ ?	372.9	365.5	367.2
How soon can DD789 reach Bermuda under normal SOA ?	28.6	17.9	23.4
Where is Admiral ____________ now ?	10.9	7.0	8.3
Are any ships scheduled to be in the projected vicinity of Typhoon Dottie in the next few days?	361.3	357.9	358.7
Aircraft or ship, with doctor aboard, nearest in time to point x, y ?	118.7	97.4	100.0
What ships with long-range radar could be in area__________ by 2400 tomorrow?	115.9	90.0	97.7

Figure 1-6. Timing of Representative Sea Surveillance Questions (milliseconds query processing only).

The availability of the associative memory had little effect on the overall design of the data files, even though it had a significant effect on the design of data records. BECAUSE THE DISC STORAGE UNIT SELECTED FOR FILE STORAGE WAS RELATIVELY SLOW, AND IMPOSED FORMATTING CONSTRAINTS, THE AVAILABILITY OF THE AM WAS LESS SIGNIFICANT IN THE OVERALL DESIGN OF FILES THAN WERE THE PECULIARITIES OF THE 818 DISC. An AM can, however, be expected to influence overall file design in other situations, and a research project in this area is recommended in Paragraph 1. 3.

The language of programmers is changed by the availability of an AM. Terms like "response vectors," "tag memory," and the other words and phrases of the AM technology become current. Surprisingly, no basic changes are made in the programmer's idea of how to make flowcharts and how to organize algorithms. THE EXISTENCE OF THE AM DID NOT REQUIRE CHANGES TO FLOWCHARTING CONVENTIONS OR TO THE STATEMENT OF ALGORITHMS. Programmers used flowcharting symbols and various techniques according to their own preference, with no noticeable need to change these basic tools.

Because implementation experience is required to test our ideas about associative processing, problems should be implemented for the GAP hybrid as soon as it becomes available. The SEA SURVEILLANCE PROBLEM IS, HOWEVER, TOO LARGE FOR A FIRST CUT AT IMPLEMENTING AN ASSOCIATIVE PROCESSOR PROBLEM. Some manageable subproblem of potentially high yield could be selected for implementation, however.

1. 3 CONCLUSIONS AND RECOMMENDATIONS

1. 3. 1 Introduction

The purpose of this paragraph is to present certain conclusions and recommendations in a summary form. These conclusions are noted in capital letters and underscored. Volume II and other paragraphs of this volume provide supporting information for these conclusions and recommendations. The subject items of this paragraph will be considered from these viewpoints:

(1) The sea surveillance problem and the hybrid associative memory system.

(2) The Goodyear Associative Processor.

(3) [illegible]re Efforts.

1.3.2 The Sea Surveillance Problem and the Hybrid Associative Configuration

By definition, there are three main components in a hybrid associative memory system:

(1) The central processor.

(2) The associative memory.

(3) Peripheral storage devices.

These three components may be considered as axes in an overall system; that is, every component may be varied to some extent. A convenient way of showing how these may be varied is to develop a recommended hybrid system for the sea surveillance problem.

The sea surveillance problem is a problem which requires the manipulation of a large data base. The volume of data requires the use of a peripheral storage device; the order of data required demands the random access features of either a drum or a disc. From a programmer's viewpoint, the choice of drum or disc is important if the data processing required must wait for data to be accessed and stored in the processor. What is desired is a system balanced timewise between data access time and processing time. In this respect, a disc file similar to THE IBM 1302 DISK UNIT IS RECOMMENDED for problems of the sea surveillance type.

The selection of a central processor may be made in an evolutionary fashion, that is, by first supposing that the associative memory did not exist. (The later addition of the associative memory processor will not detract from this initial selection of the central processor.) Because of the voluminous amounts of data manipulation required, THE BEST CENTRAL PROCESSOR FOR THE SEA SURVEILLANCE PROBLEM HAS THESE FEATURES:

(1) GATHER READ, SCATTER WRITE INSTRUCTIONS.

(2) CORE-TO-CORE MOVE OF BLOCK SIZED DATA.

Notice that this does not include the CDC 1604-B. Therefore, THE CDC 1604-B IS NOT THE BEST CENTRAL PROCESSOR that could be chosen.

The selection of an associative memory requires:

(1) A decision to determine whether the associative memory will be beneficial.

(2) A determination of the required features of this memory.

The first requirement may be answered in the affirmative; that is, AN ASSOCIATIVE MEMORY PROCESSOR IS AN ASSET IN EFFECTING problems of the sea surveillance type. This follows from the decrease in programming analysis and encoding required because of the relaxed data format and ordering constraints needed in a hybrid system. The choice of the associative processor depends upon its interface features with the central processor and its ability to manipulate data. In this respect, THE GOODYEAR ASSOCIATIVE PROCESSOR SHOULD NOT BE USED (IF A CHOICE MAY BE MADE) FOR PROBLEMS OF THE SEA SURVEILLANCE TYPE, for these reasons:

(1) Difficulty in data-dependent operations.

(2) Lack of interrupt on halt instruction.

(3) Cannot take advantage (because of the preceding two reasons) of multiprocessing aspect.

(4) Special channel not warranted — adds to cost.

(5) Sophistication (for example, luxury search instructions) not required.

The best associative processor is not described explicitly in this report. It is one processor using the interface method of another processor; that is, THE BEST ASSOCIATIVE MEMORY PROCESSOR FOR PROBLEMS OF THE SEA SURVEILLANCE TYPE HAS THE FOLLOWING FEATURES OF THE A2 PROCESSOR AND THE INTERFACE OF THE A3 PROCESSOR:

(1) TAG BITS.

(2) RESPONSE STORE MANIPULATION.

(3) PREFIX REGISTER.

In comparing two systems which differ in that one has an associative memory and one does not, it may result that processing time requirements will be of the same magnitude. In this study, it was found that the GAP hybrid system was indeed slower by about 5-10 percent than the non-hybrid system. This result was generated by the fact that the interface arrangement required that data be transferred between the disc and the associative processor through the 1604-B memory. If the AS interface arrangement is employed, the time requirements will not be improved significantly on an overall processing basis since the data access times remain constant. Thus, a decision of choice between the two systems should not be based on time requirements alone. Even at the sacrifice of some processing time, it was found that EVERY HYBRID SYSTEM STUDIED RESULTED IN THESE ADVANTAGES:

(1) REDUCES PROGRAMMING ANALYSIS AND ENCODING WITH RESPECT TO DATA ORDERING.

(2) ADMITS POWERFUL POTENTIAL IN PROCESSING QUERIES.

(3) PERMITS SIMPLIFICATION OF DATA BASE MAINTENANCE PROCESSING.

It should be remembered that the sea surveillance problem used in this study is not the operational sea surveillance problem, because of security restrictions; the problem considered may bear little resemblance to the real problem. Additionally, this study has designed a system that is biased towards the hybrid system composed of the CDC 1604-B central processor, the CDC 818 Disc File, and the Goodyear Associative Processor. For these reasons, IT IS NOT CERTAIN THAT AN ASSOCIATIVE PROCESSOR SHOULD BE USED FOR THE OPERATIONAL SEA SURVEILLANCE PROBLEM.

After many systems are designed, a natural evolutionary process is to encode these systems to obtain empirical results. It is felt that the system developed in this report is much too large for such an undertaking. Therefore, IT IS NOT RECOMMENDED THAT THE ENTIRE SEA SURVEILLANCE SYSTEM GIVEN IN THIS REPORT BE ENCODED TO ACHIEVE EMPIRICAL RESULTS. On the other hand, the system reported on contains many interesting subsystems which are also useful in other applications. For example, the query language translator (Paragraph 7.2.1, Volume II) consists, in the main, of processing which converts a query language statement into a canonical Polish prefix form; the query pre-processor accepts this Polish prefix form and "compiles" it into a command list; the query run routine accepts the command list as input and performs it in an interpretive fashion. These subsystems in themselves are applicable to problems other than sea surveillance. Therefore, IT IS RECOMMENDED THAT SOME SUBSYSTEM BE ENCODED FOR THE CDC 818 DISC FILE, CDC 1604-B CENTRAL PROCESSOR, AND THE GAP TO OBTAIN EMPIRICAL RESULTS.

1.3.3 The Goodyear Associative Processor

Before any encoding is started, IT IS STRONGLY RECOMMENDED THAT A COMPREHENSIVE PEDAGOGICALLY SOUND PROGRAMMING MANUAL BE WRITTEN FOR THE GOODYEAR ASSOCIATIVE PROCESSOR. This manual should contain at a minimum those items noted in Paragraph 3.3.2.

It is perhaps too late in the development of the GAP to make many (if any) changes in the logic. However, several such changes may be recommended for the first and/or future processors. These changes are listed in a utility order:

IT IS RECOMMENDED THAT THE GAP BE MODIFIED TO:

- PROVIDE AN INTERRUPT TO THE 1604-B AFTER EXECUTING THE HALT INSTRUCTION.
- INSERT THE ADDRESSES AND COUNT OF RESPONDERS INTO A 1604-B CORE MEMORY WITHOUT ZERO FILLING.
- PROVIDE THE ABILITY TO READ DIRECTLY FROM PERIPHERAL STORAGE INTO GAP.
- PROVIDE INDIRECT ADDRESSING AND/OR INDEX MODIFICATION OF ASSOCIATIVE MEMORY ADDRESSES.
- PERMIT READ/WRITE OPERATIONS OF ELEMENTS OF THE D AND E REGISTERS.

The first two modifications are most important. If effected, such changes would enhance the multiprocessing features of the system by permitting effective control (the Halt instruction — interrupt change) and removing a need for intermediate 1604-B operations to form subsequent instructions to load (or unload) the memory, determine data flow, etc. The remaining changes would enhance the GAP if they were effected but these are not essential changes. It should be noted that the first recommended change is the most important, the easiest to effect*, and the least expensive.

1.3.4 Future Studies

Throughout the course of this study, several questions arose which could not be answered directly, if at all, because of time requirements. These questions were of such a complex nature that the answers desired warranted a separate study of their own. These questions then are the genesis for recommended future studies. The following paragraphs have attempted to form each study as an independent investigation.

1.3.4.1 Other File Structures. The current study focused attention on one particular file structure: a multi-list file structure in which the links usually contained in data records are extracted and placed in directories. Other file structures should be investigated to determine the effectiveness of AMs for different file structure types. Some examples of file structures are:

(1) Fixed field, fixed size data records.

(2) Variable field, variable record size.

(3) Fixed field, variable record size.

(4) Files with records embedded within records.

For each type of file structure the problems of query, maintenance, and data base loading should be investigated. The effectiveness of an AM for problems with such files could be assessed, and the importance of the AM interface and guidelines could be determined to indicate when an AM approach would improve the data processing for each file type. A trade-off study should be conducted to indicate the point where the AM utility is important. For example, if record sizes for a fixed field, fixed size

*A possible method for accomplishing this change is to interchange the interrupt caused by the parity error light and the resetting of the AM active line by the Halt instruction. This may be as simple as the interchanging of two wires.

data record were such that only one data record could be placed into an AM, then the AM would not be an effective device for such problems since a fixed record obviates the "associative" need of the AM. On the other hand, if the record data description were placed in AM, the programs could be written in an interpretive fashion and querying the AM would isolate the precise location of the data in the record.

Investigation or implementation of a comprehensive system would not be appropriate for this type of study. More would be accomplished by developing test problems and producing careful timing analyses. The study should analyze the variant AMs investigated during this current effort. If one AM is considered, it should be either the GAP or, preferably, the A2/A3 configuration. The advantage of choosing test problems judiciously is that they can actually be implemented in an AM without great expense. Much will be learned about AM techniques when one actually exists and is utilized for experimentation. Depending upon the results obtained from such a study, a specific application of a data base of importance to intelligence could be implemented. This would, however, be a second phase to this study rather than a first phase. <u>A RECOMMENDED STUDY IS TO EXAMINE THE IMPACT OF AN ASSOCIATIVE MEMORY (EITHER THE GAP AND/OR THE A2 LOGIC WITH THE A3 INTERFACE) UPON THE NOTED FILE STRUCTURES.</u>

1.3.4.2 <u>Executive Control Programs.</u> In this study, several executive type programs were required. For example, the query controller and query input function to control the allocation and input of data from the disc file into the core memory; the AM Dispatcher determines and effects the segmentation and processing control of a functional subroutine and its inputs, outputs, etc. In all cases, data was transferred from the disc file into the associative memory through the 1604-B core memory. As noted in this study, to take advantage of the A3 configuration a centralized program must control input-output fro core to disc so that, under instructions from the programmer, data may be sent directly to the associative memory (or from the associative memory to disc) without wasting a transfer to another section of core. From these examples, it is clear that consideration must be given to the executive control program which resides in core to enhance the system use of an AM. The executive control program must further be able to analyze error conditions that arise from the associative memory so as to be able to take corrective action whenever possible.

The study noted in the previous paragraph is in itself not ver· large. It should, however, be broadened to investigate how an executive contro. program might take advantage of an associative memory. Certain tables required by the executive control program might reside permanently in AM (e. g., memory maps of core and disc which show the available space as required for dynamic memory allocation). Speeding up table lookups could decrease the overhead time of executive control programs. On the other hand, the increased cost of using an AM, and the decreased AM space for other processing functions may not be warranted. Trade-off studies should indicate the importance of evaluating the use of AMs for executive control programs, and more broadly for the use of AMs in a multi-programming environment.

IT IS RECOMMENDED THAT EXECUTIVE CONTROL PROGRAMS WHICH SERVICE AND USE THE ASSOCIATIVE MEMORY AND COORDINATE MULTIPROCESSING BE INVESTIGATED.

1.3.4.3 The A2/A3 Configuration. As noted (Paragraph 1.3.2), based upon this study, it would appear that the A2 configuration as interfaced with the central processor in A3, would be the most advantageous for problems such as sea surveillance. The current study did not permit sufficient time to explore this combined configuration. Before such a configuration is implemented, further studies, such as listed below, should be made.

(1) Detail the logic associated with the A2/A3 configuration and develop a programming manual which can be utilized by programmers.

(2) Develop comparative timing of the A2/A3 configuration for the sea surveillance problem.

(3) Implement test problems for the A2/A3 co[illegible]iguration other than the sea surveillance problem. For example, RADC has sponsored work on other problems for the GAP configuration. These problems should be coded for the A2/A3 configurations. If an advantage were indeed evidenced, it would then be appropriate to sponsor implementation of the A2/A3 configuration oriented towards an appropriate central processor

IT IS RECOMMENDED THAT THE A2/A3 HYBRID CONFIGURATION BE INVESTIGATED TO THE SAME LEVEL OF DETAIL AS THE GAP, A2, AND A3.

1.3.4.4 Alternative Associative Memory Configurations. In developing the other (A2 and A3) associative processors considered in this report, other processor configurations were considered. Several of these appear to have special promise and interest and should be considered for study. These are:

(1) A multiprocessor with full parallel logic.

(2) A multiprocessor with response store in thin film.

(3) An integrated system with response store in thin film.

(4) A special input-output or search controller with full parallel hardware.

(5) A special input-output or search controller with response store in thin film.

The studies would be both hardware and software oriented. The development of the technology to achieve a response store in thin film will be primarily hardware oriented. This technology could permit greater manipulation of tag bits at a relatively inexpensive cost. Use of an AM as a special I/O controller could decrease input-output time, which is a major part of the sea surveillance and other command and control type problems.

IT IS RECOMMENDED THAT THE ABOVE STUDIES BE INSTITUTED BEFORE ONE VENTURES OUT TO UTILIZE AN ASSOCIATIVE MEMORY IN A SIGNIFICANT APPLICATION. A great deal can be gained from paper studies at a smaller cost than implementing a complex application. The recommended procedure study is to:

(1) Perform paper design analyses.

(2) Specify the features of a small AM for general utility.

(3) Develop the AM.

(4) Develop programming tools.

(5) Implement simple, well-defined problems and make comparative studies.

(6) Utilize the AM-hybrid system for applications.

1.3.4.5 Compiler Study. To take advantage of an associative memory, the programmer should have available a language in which he can write his programs conveniently. No programming language currently permits programmers to write statements for an AM in a language and have these statements compiled to generate object code.

With the present state of our technology it is not feasible to permit the programmer to write statements independent of whether the data is in core or AM (this applies equally to A3 where AM functions as a conventional memory). Whether a data element should be processed in core or in AM can be decided only by the programmer. To attempt to develop a program which does not require the programmer to specify the data locations is a major research effort which may have no solution and is therefore not recommended. The study should be conducted for the alternative configurations studied in this report and for the recommended A2/A3 configuration.

<u>THE MAIN AREAS OF THE RECOMMENDED STUDY AND INVESTIGATION ARE THE DESIGN OF THE COMPILER/INTERPRETER, ANALYSIS OF MULTIPROCESSING CAPABILITIES, AND EVALUATION.</u>

The following specific tasks should be accomplished in each part of this study:

(1) Design of Compiler/Interpreter

(a) Determine modification, if any, to a typical language required to effectively utilize the associative memory.

(b) Determine the functions to be performed by the associative memory.

(c) Determine whether the compiler/interpreter should be a compiler or an interpreter, or both.

(d) Determine necessary language extensions required to permit the language user to control associative memory functions (e.g., a declarator statement which specifies that a block of code is to be AM code, or conventional code is desirable. The statement would have the form (CODE IS $\left\{\begin{matrix}\text{AM} \\ \text{STANDARD}\end{matrix}\right\}$). A corresponding delaration having the function of putting data into AM might have the form DATA IS $\left\{ \begin{matrix} \text{STANDARD} \\ \text{AM} \end{matrix} \right\}$.

Additionally, statements to:

(i) IDENTIFY

(ii) FETCH

(iii) ERASE

(iv) SUBSTITUTE

(v) INSERT and

(vi) CHANGE would be required as a minimum.

(e) Determine the most effective techniques for utilizing associative memories and features of an associative memory that would enhance the compiler/interpreter process (e.g., determine whether the AM should be utilized during the compilation phase and how it would be utilized).

(2) Multiprocessing Capabilities. Review existing control program for a typical processor to:

(a) Determine modifications required to extend executive control over the associative memory.

(b) Determine the interdependence between the compiler/interpreter and the executive control during compile and/or run time.

(c) Determine the additional features required by the control program to aid in the manipulation of an object code developed by the compiler/interpreter.

(3) Evaluation

(a) Comparative evaluation of alternative AM interfaces for the compiler/interpreter processes.

(b) Extended language developed in Bachus Normal Form.

(c) Programming Manual.

(d) System design of the most effective compiler/interpreter.

(e) Effective techniques for AM manipulation.

(f) Recommended changes to the executive routine and substantiated reasons.

If warranted, an implementation phase could be instituted.

1.3.4.6 Investigate Large Associative Memories. The findings and conclusions reported on in this study do not necessarily apply to large associative memories. The technology associated with large AMs will certainly be substantially different from that of small AMs that exist today. Several problems must be studied with respect to large AMs. These include:

(1) How will the organization of computers be affected by AMs? As noted in GAP, data-dependent processing is restricted by transfers to and from GAP and conventional core.

(2) What should the instruction repertoire of such systems be? By having programmers involved in the early stages of such investigations, some of the errors made with GAP (assuming that sea surveillance is the problem to be implemented) can be avoided.

(3) What applications will a large associative memory assist? There have been several papers which conjecture that a large AM will be effective for such problems as data storage and retrieval, natural language manipulating and fact retrieval. It is not entirely clear that a large AM will make significant impact on such problems. Currently, work in natural language manipulation and fact retrieval requires more efforts in linguistic analysis in which associative memories certainly cannot help.

SECTION II. DISCUSSION AND COMPARATIVE ANALYSES OF ASSOCIATIVE MEMORIES

2.1 ASSOCIATIVE MEMORIES

Every item stored in the memory of a computer system may be considered as a pair of numbers: a data word and an address. The data word may be interpreted in many different ways: as a binary number, a decimal number, an alphanumeric character or other symbol, a command, any part of any of these things, or a combination of several of them, either homogeneous or heterogeneous. The address refers to the physical device in which the data word is stored. It may be accessed directly or indirectly, relatively or absolutely, with different groupings of storage elements possible, as in variable word length.

These two numbers are usually handled very differently. In the vast majority of applications, data words are manipulated, arithmetically or logically, in the program, and the results located in memory by their addresses. The data words may themselves include addresses, and great flexibility is afforded by the manipulation of addresses in the program. But even in sophisticated routines involving indexing, the ultimate result is usually the retrieval of data words from specified memory locations at the right times in the program. And each data word is usually read from memory with no regard whatsoever for its content.

The dissolution of the barrier between data and addresses can add greater flexibility to a system's capability. When memory is accessed according to its contents, certain kinds of decisions are greatly facilitated. Addressing by content may be employed in either writing or reading. Words may be placed in memory according to the nature of the data they include, so that the physical address implies something about the content. Or words situated arbitrarily may be retrieved according to their content. (The human memory uses both of these techniques extensively and sometimes inexplicably, e.g., "I'll remember the data George Washington was born because it has the same digits as $\sqrt{3}$," or, "I don't remember his name, but it begins with a Z - oh, yes, Fitzpatrick!")

An "associative" or content-addressable memory can be either a software or a hardware item. The circuits in a memory device may be specifically designed to facilitate addressing by content. On the other hand, any of the operations performed on or by such a memory can be imitated by a program using a general-purpose computer with a conventional memory. There can be tremendous disparity in the time and space requirements for the operation. This report discusses some of the tradeoffs involved in the choice of how to implement associative-memory operation.

Location in memory by addresses is generally unique; that is, each address corresponds to one and only one data word. Addressing by content implies the consideration of more than one data word at a time. A number of possible candidates is examined, and checked as to their content, against certain criteria. In many cases, the criteria will not apply to exactly one number, but may be met by many or by none. For this reason, the basic operations in an associative memory are not only transmissions of data, but partitions of the items into sets according to the outcomes of various interrogations. Each test results in a dichotomy, dividing the words into sets of "responders" and non-responders. One or all of the responders may be retrieved and manipulated after a test. Or the sets defined by the various tests may be combined by union and intersection into other sets that are of interest. These processes call for one or more registers to identify sets of numbers, with one bit in a register for each word in the associative memory. Logical operations on the contents of such registers constitute an important part of the operation of the AM. The results of these operations in turn control transfers of data, by means of such operations as "write in all responders" or "read first non-responder."

Most of the comparison operations make use of one or more reference numbers or comparands. In AM hardware, a comparand register is provided, with the capability of simultaneous comparison with all items in a portion of the AM. The operation may be bit parallel or bit serial; the latter is well-adapted to the logical requirements of most comparison operations. A second register accompanies the comparator register for masking. This adds the important facility of ignoring some bits in a comparison. It is especially useful in the many applications where data words consist of several different parts with their own codes and meanings.

The basic structure of an AM is as shown in Figure 2-1. The memory proper holds W words of B bits each. The comparand and mask registers have B bits each, and the response register(s) W-bits. Parity bits may be added to any of these registers. There may also be one "busy bit" per word to indicate whether it contains significant data. The busy bits constitute a specialized response register; the set of words it identifies is usually ANDed together with a set of words satisfying search criteria to form a W-bit response

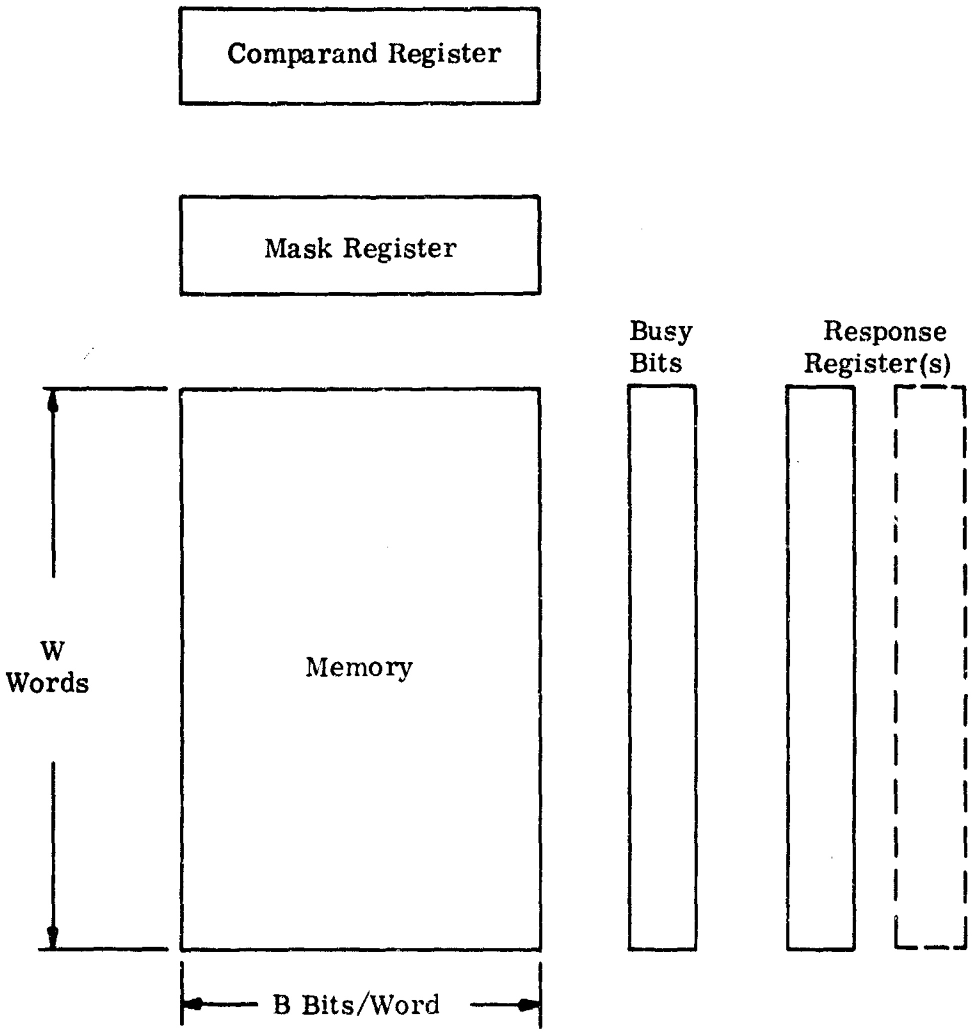


Figure 2-1. Basic Structure of AM

word. That is, the intersection of the set of responders and the set of "busy" words is used as the response word. The register of busy bits has inputs to allow the modification of these bits during loading and unloading of the memory. Additional busy-bit registers could be added for further refinements.

The basic operations on the contents of the AM are identifications of sets of numbers that pass certain tests. A word is set up in a response register containing a ONE for each word that is a member of the set and a ZERO for each word that is not. The criterion for membership in the set is usually a relationship with the comparand: equal to, greater than, or less than. There are also operations that do not depend on a comparand, such as maximum or minimum. These operations often identify a one-membe set, but there may be more than one location in memory with the same value..

The basic operations are easily combined into other tests, such as unequal, greater than or equal to, between two limits, next greater than, congruent modulo M, etc. These combinations may all be effected by logical operations on response words, but a number of them may be built in as single operations, depending on the hardware's facilitie

The contents of response registers are used in two ways. They are logically combined with each other to form new response words. And they are used to control trans fers into or out of the AM. A set of numbers specified by a response word may be read, written, or deleted. Reading or writing may be block transfers between the AM and the computer's active memory, or may be carried on successively, with operations performed on each word.

The logical combinations of response words are limited by the number of respons registers available. But even if there is only one response register, a certain amount of logical manipulation is possible. The results of any test may be ORed or ANDed with the previous contents of the response register, as a part of the procedure. Multiple registers permit the storage of both words, as well as the result of the combination.

The response registers may or may not make their contents available to the computer. They may also be capable of producing a count of the number of ONEs they contain. This count is needed during the course of some operations. In some cases it is not a full counting capability, but a three-valued count, to show whether the register contain no ONEs, exactly one, or more than one.

This report discusses some of the many possible configurations of an associative memory, with particular emphasis on the different ways they may be integrated with the 1604-B computer. These range from treatment of the AM as a peripheral device to a total integration of the AM into the 1604-B memory. One of the systems is Goodyear's GAP. The others were developed by AUERBACH to show the effects of changes in the design.

2.1.1 Goodyear GAP

The Goodyear associative memory, GAP, has a capacity of 2,048 words of 48 bits each, plus parity and busy bits (see Figure 2-2). Beside the busy bits, there are three response registers of 1,024 bits each. Each operation involves either the upper 1,024 words or the lower 1,024 words. Results of the search (ANDed with the busy bits) are formed in the response store, which is called the S register. This, in turn, communicates with the other two 1,024-bit registers, D and E. The usual practice in searching the whole memory is to store the results for one half of the memory in each of these two registers, but this can be varied at the programmer's option. Contents of the S, D, and E registers can undergo a limited number of logical operations.

The numbers in the S, D, and E registers are not directly available to the 1604-B. A response resolver examines the contents of the S register and produces either a count of the number of responders or a 10-bit address corresponding to the location of the first ONE. These numbers are used to control subsequent communications between the AM and the 1604-B.

The principal inputs into the AM from the 1604-B are:

(1) Data words for the comparand and mask registers.

(2) Data words to load the AM.

(3) Addresses for AM loading.

(4) Search instructions.

The first three of these inputs are transmitted as 48-bit words. The search instructions use the 1604-B's nine-bit external function code instruction. Composite instructions of more than nine bits may be transmitted as data words, and stored in a separate register within the AM. Outputs from the AM to the 1604-B are data words, including the results obtained by the response resolver, and various control signals.

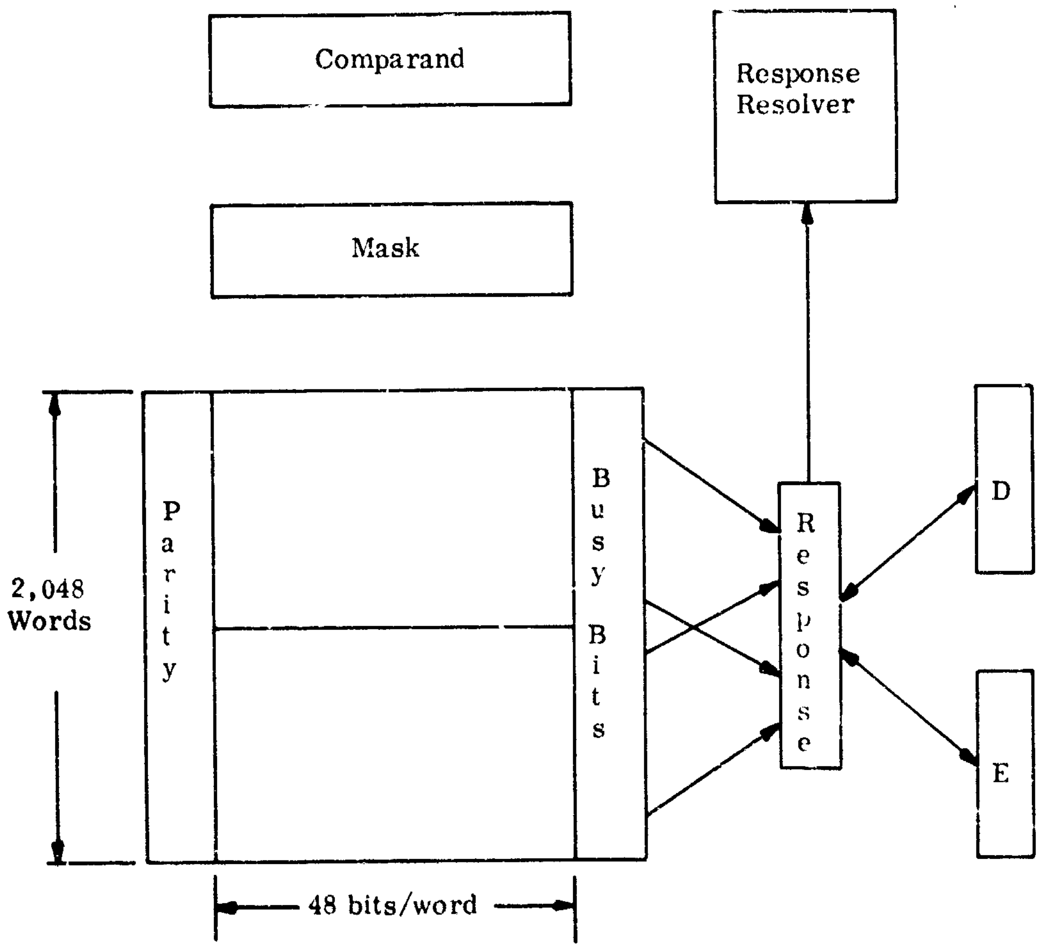


Figure 2-2. Goodyear GAP

The central feature of the GAP memory is the BILOC element used to hold each bit. The fast response of these elements makes it possible to scan the entire half memory in five microseconds, comparing 1,024 bits simultaneously with each of the 48 bits of the comparand and forming the results in the S register according to any of several search algorithms. A test for equality or inequality (exact match or mismatch) is performed in one pass. A more involved test, such as "between two comparands," is executed by a sequence of tests, with logical manipulation of the results in the S, D, and E registers. A search for the maximum of all the numbers in half the memory may be performed in one pass or several, depending on the technique used. Some searches take a variable number of passes, depending on the values that happen to appear.

The operations that take more than one pass are performed within the AM after one command from the 1604-B. The most involved operation that can be performed in this way is the complex search, in which the comparand is divided into as many as eight sections, and successive searches, which may be of different kinds, are carried out. The results may be ORed or ANDed together during the operation. Such an operation may carry out a composite search in response to one program step; for example. "Identify all words that contain a 5 in index position X, and have a value between 0.489 and 0.511 in columns j to k, except those with a 0, 1, or 2 in index position Y; also include any word with a negative number in position Z, or with a value of at least 21 in columns 1 to m; and from among these select the word with the lowest number in columns n to p; in case of a tie, pick the word with the highest value in columns q to r." There are restrictions as to the arrangement of the various keys, and masking cannot be used in the usual way in complex searches. But any number of complex searches may be combined by successive instructions from the 1604-B.

The GAP memory thus operates as an independent unit, carrying out its searches and then making results available to the 1604-B. The program may call for the transfer of the responding words into core storage, or the reading of their addresses, or the count of how many responders appeared. Alternatively, there may be a command to replace the responders, or simply to erase them without any transfer from the AM to the 1604-B.

The interface between the AM and the computer is assumed to use Channel 7, the 1604-B transfer channel. Several variations on this idea have been proposed by Goodyear. They discussed a spectrum of possibilities, ranging from treatment of the AM as a strictly peripheral device to a tight integration providing the AM with direct access to the 1604-B memory. The various possibilities in between the extremes involve modification of one, two, or three registers in the 1604-B. Without modification, a load, unload, or search operation may have to be preceded by successive transfers to load the necessary registers in the AM. With modifications to one or more of the A, Q, and B registers, this set-up time is reduced. With direct memory access, the execution times for the operations are also reduced. But in all of these configurations, the AM operates asynchronously with the 1604-B, so that each operation is likely to involve waiting times at the beginning and end, and for memory access in between.

2.1.2 Configuration A2

Another combination of AM hardware and interface to be discussed in this report is known as A2 (see Figure 2-3). It is a version that uses less hardware than GAP, leaving more operations to the 1604-B. On the other hand, it employs more sophisticated operation codes than GAP in order to facilitate the execution of the same operations with less equipment.

Configuration A2 had a predecessor in the design stage, known as A1. This was stripped down almost to the bare minimum of hardware. It retained the registers and operations that were indispensable to AM operation, with additional facilities added where they were seen to provide a considerable increase in ability at a moderate cost. In the transition from A1 to A2, this same philosophy of expansion was continued. Economical extensions, providing operational improvements that compare favorably with their cost, were included in the design of A2. Thus, A2 represents a conservatively simple associative memory.

In this configuration there is only one response register, but three auxiliary features help in the formation of logical combinations of response sets:

(1) The results of a current test may be combined with the previous contents of the S register.

(2) There are eight sets of "tag bits" which may be used to mask any search.

(3) The contents of the S register are available to the 1604-B.

The combinations of previous contents of S with current searches are not limited to AND and OR, but may be <u>any</u> logical function performed on each bit. This includes equivalence, exclusive or, if, etc.

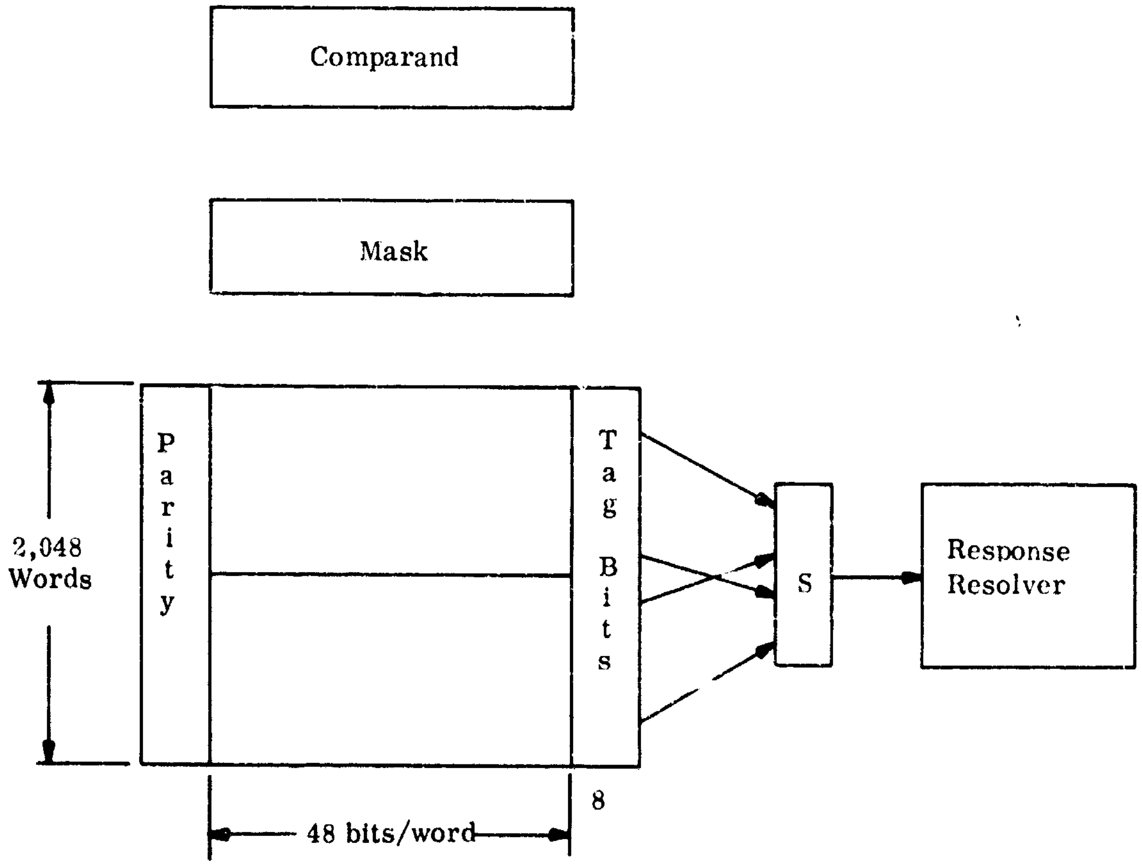


Figure 2-3. Configuration A2

The tag bits may be thought of as eight 1,024-bit registers whose contents may be ANDed with the results of any search The loading of these "registers" takes place bit-by-bit as words are loaded into the memory.

Unlike the GAP configuration, A2 offers the contents of the S register for reading into the 1604-B. The 1,024-bit word is divided into 32 words of 32 bits each for transfer into or out of the 1604-B. Transfers into the S register may include the formation of any logical combination of the previous contents of S and the word supplied by the 1604-B. Thus, the 32 words representing the results of a previous search may be combined with the 32 words of a new one as they are written into S. Any other manipulations of response vectors may be performed within the 1604-B, and the results can be placed in S to control subsequent transfers.

Loading of the memory in A2, and reading from it, are similar to the corresponding operations with GAP.

Reading and writing operations may select any one of the eight tag registers to identify different sets of words in the AM, to restrict transfers to members or non-members of those sets, and to modify the tag bits during writing.

In general, the potentiality of A2 lies in the flexibility of its operations. The programmer is presented with a broad range of possibilities in the performance of any search. If these are exploited fully, this AM will extract a maximum amount of useful information from each of its operations. Programming guidelines for this configuration are given in Section II, Volume II.

2.1.3 Configuration A3

Configuration A3 achieves the ultimate in AM-1604-B integration by making the associative memory a part of the 1604-B core memory. The highest-numbered 2,048 words of the 32,768 in the core memory are simply replaced by the GAP hardware or its equivalent. The BILOC element has read-write capabilities compatible with those of 1604-B core memory, so that each location in the AM can be addressed normally. It is accessible to the program, and for communications in both directions with whatever peripheral devices are employed in a system.

As in the other configurations, the S register is 1,024 bits, and takes the results of searches on one half of the memory at a time. There are also the D and E registers, with the capability of communication with the S register during each operation. The effect of these provisions is that the results of a current search may be ANDed or ORed with, or simply replace, the contents of S, D, or E, and the result stored in any of the three registers, regardless of which it depended on. The previous contents of the two registers not used in the comparison will be preserved in the process, but will be moved to different registers if necessary. For example, the registers may contain after a comparison:

D:	(previous contents of E) AND (result of current search)
S:	(previous contents of D)
E:	(previous contents of S)

These changes are effected by interchanging the contents of registers before and after the search.

The response resolver in A3 generates the count of the number of ONEs in D, E, or S, or the addresses of responders. The contents of S, D, and E themselves are not available to the computer in forms other than responses.

An important feature of A3 is the absence of certain registers, which is possible because their functions are executed by normal 1604-B registers. Some of these have been omitted from the preceding diagrams for the sake of simplicity. Both GAP and A2 have 11-bit address registers to select words in the AM for writing or reading. They also have 48-bit instruction registers, which are used to hold instruction words transmitted by the 1604-B (e.g., for the GAP complex search). Both of these registers are obviated in the A3 configuration. Addresses of words are like those in the rest of core. They are distinguished by the fact that their first four bits are 1111, which are sensed by the computer in order to carry out functions peculiar to the AM, such as the manipulation of busy bits or reading responders.

Instructions for the AM are part of the regular 1604-B program, and use the computer's own instruction register. They utilize the usually illegal instruction numbers 00 and 77 (octal). The remaining 18 bits of the instruction word are used to direct the AM operations. Details of this scheme have been worked out fully and are presented in Section II, Volume II.

2.2 COMPARISON OF CONFIGURATIONS STUDIED

2.2.1 Introduction

This section amplifies the different features contained in each of the four associative memories studied and compares the four hybrid associative memory 1604-B configurations. The three approaches are represented by the A1 and A2 configurations (which are basically similar), the GAP configuration, and the A3 configuration. It is difficult, if not impossible, to illustrate the differences in approach, implementation, and capability by a simple comparison chart. A full understanding of the design, intention, and function of these configurations is necessary to appreciate their different capabilities. For example, the team anticipated that the A3 configuration would best facilitate system processing because of its full integration with the CDC 1604-B. This expectation is borne out only if the A3 configuration and the 1604-B provide a core-to-core block transfer or move instruction. Because of the small associative memory, such an instruction would be invaluable in the processing involved in sea surveillance. Such a relatively small difference in configurations, i.e., the presence or absence of a move instruction, cannot be demonstrated adequately by comparison tables, but can appreciably enhance the system processing for an application. See Table 2-1 for the transfer rate of the 1604-B-A3 move instruction that can be achieved by modifying the 1604-B.

Because the A2, A3, and GAP configurations each represent a different organizational approach to the employment of identical technology, and because they have greatly different inherent capabilities and their costs are radically different, each should be evaluated separately against different applications. Only in the light of a given application can one be called better or worse than another. The comparison of these configurations applies only to the sea surveillance type of problem. Evaluation against applications would also serve to discover minor shortcomings (such as the move instruction for the A3 configuration) whose removal would enhance a given configuration in a given application.

2.2.2 Physical and Organizational Differences in the Hybrid Configurations A2, A3, and GAP

Some physical and organizational differences in the four configurations are displayed in Table 2-1. This paragraph elaborates upon the differences and, where relevant, relates the effect of the difference upon the sea surveillance problem.

TAI [illegible] ION CHARACTERISTICS

ITEM	A1	A2	A3	GAP
1. Number of General Purpose Response Vector* Registers	1	1	3	1
2. Total Number of Response Vector Registers	1	1 plus 1/32 used internally plus 1604-B Core plus 8 in tag bits. (slow)	3	3
3. Flip-Flops per Memory Word in Response Store	1/2	17/32	5/2	5/2
4. Response Vector Shift?	No	Yes (+ or -) relatively slow	yes (+)	yes (+)
5. Logic on Response Vectors?	No (AND Equal search)	All (slow)	AND, OR (general)	AND, OR (restricted)
6. Use of Response Store?	Mark addresses for retrieval	Mark addresses or words for retrieval	Define a general base execution address	Mark words or position numbers for retrieval
7. Step-by-Step Processing of Responders?	Yes	Yes	Yes	No (must block transfer to 1604-B)
8. Number of Hardware Searches	4	4	14	11
9. Number of Search Instructions	4 plus modifier fields	4 plus modifier for logic	14 plus modifier fields	11 plus modifier fields
10. Number of Direct Data Manipulating Instructions	4	13	56 plus modifier fields	5 plus modifier fields

* Amdahl, G.M., et. al. Architecture of the IBM System/360 IBM Journal of Research and Development Volume 8, no. [illegible], April, 1964.

TABLE 2-1. CONFIGURATION CHARACTERISTICS (Contd.)

ITEM	A1	A2	A3	GAP
11. Number of Housekeeping & Load/Unload Instructions	22	34	24 plus modifier fields	27 plus modifier fields
12. Number of Tag Bits	1	8	1	0 (has busy bit)
13. Tag Control on Loading/Search	yes	yes	yes	no
14. Source/Sink for Load/Unload	1604-B core	1604-B core	Any peripheral device or 1604-B normal core	1604-B core
15. I/O Load (1604-B Cycles)/Word Trans. (in blocks or singly)	1/1-1/2	1/1-1/2	2/NA	1/2 or 3
16. AM Instructions per 1604-B Word	2 (EXFs)	2 (EXFs)	2	1
17. Comparand and Data Shifting?	no	no	yes	yes
18. Mask Control on Transfer/Search	no	no	yes	no
19. Shift Control on Transfer/Search	no	no	yes	no
20. Resolver Resettable	no	no	Yes, also automatically for I/O transfers	no
21. Configuration	Peripheral device on Channel 7	Peripheral device on Channel 7	Fully integrated in 1604-B structure	Peripheral device with independent pgm. sequencing on special DMA Channel.

TABLE 2-1. CONFIGURATION CHARACTERISTICS (Contd.)

ITEM	A1	A2	A3	GAP
22. Cost Relative to GAP (including 1604-B modifications but not 1604-B or peripheral equipment)	59%	62%	95-105%	100%
23. Special Comment	A1 is a stripped-down machine for study purposes and training only	A2 is about the minimum workable configuration which is balanced for the AM technology employed	A3 retains all normal 1604-B functioning and adds associative features. A MOVE (block transfer) instruction in the 1604-B would be very valuable	GAP is an ambiguous configuration — not yet a multi-processor, a little more than a peripheral device. Much better coordination and synchronization facilities are needed. The design should be more efficient using 1604-B memory and should allow more detail control by the 1604-B

(1) Number of general-purpose response vector registers. This relates to the number of 1,024-bit registers which can receive the results of a search in which processing of response vectors can be performed. There are three such registers in A3 while the other configurations have one each. The programmer has greater flexibility with A3, although for sea surveillance the lack of such registers would not be critical.

(2) Total number of response vector registers. This relates to the total amount of storage in units of 1,024 bits provided for response vectors. GAP and A3 provide three such registers; A2 provides one register plus 1/32nd of a register which is used internally to accomplish processing of response vectors albeit slowly and further has provision for the storage of eight tag bits; A1 has a single response vector register.

(3) Flip-flops per memory word in the response store. A3 and GAP have two and one-half flip-flops per word; A2 a little more than one-half flip-flop per word; and A1 one-half flip-flop per word. As shown in Table 2-1, the response store complexity primarily makes up the difference in costs between the several memories.

(4) Response vector shift. This indicates whether or not a response vector can be shifted to allow the processing of fields and adjacent words in a logical manner. This is particularly important for problems such as sea surveillance since data words manually extend over more than one word. A1 does not shift (and hence is severely limited); A2 shifts up and down, but does this slowly; A3 and GAP both shift down at high speed. It was found that the ability to shift up and down would be useful to the programmers.

(5) Logic on response vectors. This indicates which of the 16 possible functions of two binary variables can be applied on a bit-by-bit basis between a present search and previous results, or, in some cases, between stored response vectors. A1 provides no logic per se but does provide for ANDing an equal search with previous results; A2 provides for all logic functions to be processed in 32 steps and,hence, provides them slowly; A3 provides a general AND/OR capability among the three response vectors plus additional setting and resetting capabilities; GAP provides a restricted AND/OR function between results of a current search and previous results. The capability of the A3 is the most flexible. The cost of providing it with the GAP technology is not significant and should be provided.

(6) Use of response store. This indicates the use the programmer may make of the results of a search or sequence of search stores in a response vector. In A1, the response store only conditions 15-bit addresses for retrieval from A1; in A2, the response store conditions the retrieval of addresses or words stored in the AM; in A3, the response store serves to define a 15-bit base execution address which may be used in any manner available to the 1604-B (including the immediate retrieval of addresses or data, indexing, or indirect addressing); in GAP, the response store marks words for retrieval or (because no prefix register is employed) positions numbers in AM for retrieval.

This restriction in A1 makes it useless for the sea surveillance problem. It implies that an image of the data in AM exists in core memory. Since changes in AM require corresponding changes in the core image, a great deal of wasteful processing results. The flexibility provided by A3 is exceedingly useful for the programmer. The ability of the programmer to fully manipulate the response store is essential since this is the only interface between the programmer and the AM data. By its very design, the power of the AM lies in its capability to search on contents and perform operations on lists.

(7) Step-by-step processing of responders. This indicates whether or not the 1604-B can process responders individually and in order from the AM, or whether it must retrieve a block of responders and perform its own program loops. A1, A2, and A3 allow step-by-step processing, while GAP does not. The flexibility found in A1, A2, and A3 is more desirable than that of GAP. The programming sequence for the former is:

SEARCH

PROCESS

GET NEXT,

while the programming sequence in the latter is:

SEARCH

READ INTO CORE

SET UP LOOP

PROCESS

GET NEXT.

As can be noted, with GAP, one is working essentially with two different memories, GAP and core.

(8) Number of hardware searches. This indicates the number of different searches for which the hardware and sequencing are provided in AM. A1 and A2 implement four different searches; A3 implements the 10 basic searches of one GAP plus four slow variations of max, min, next higher than, and next lower than; the GAP implements the 10 basic searches plus the between limits search. The searches in A1 and A2 are equality, greater than or equal, and equality-AND, greater than or equal-AND. The latter two searches cost almost nothing since no per-word hardware is added. The only additions are a couple of control flip-flops and decoding gates. In the work performed for sea surveillance no particular advantage was found. The addition of tag bits, as in the A2 configuration, should be adequate for saving response store vectors. The search instructions represent the power of the AM and are useful. Table 2-1 shows the use of the search instructions on several subroutines coded for GAP. Of the searches, programmers felt that it would be just as convenient to program the NHC and NLC rather than to include them as hardware features.

The A3 search is more general than GAP. In the GAP, the D and E registers are associated with the upper and lower halves of memory; in A3, generality is provided through the detail control of a search available in A3.

GAP contains one search feature not contained in any of the AUERBACH memories. This is the complex search which permits the user to segment a word into fields and specify for such field the type of search required. This is a powerful instruction when operating on tables where different fields exist. However, no advantage was taken of the instruction for sea surveillance since data fields usually extended over word boundaries. The complex search can be provided as a macro with more capabilities than provided in GAP. The macro could permit the programmer to jump out of the search on a particular field to do processing and then return to the search.

(9) Number of search instructions. This indicates the number of instructions which directly cause or control searching of AM. Each configuration provides one basic instruction for each search plus modifier fields. The modifier field control is most extensive and powerful in A3.

(10) Number of direct data manipulating instructions. This represents a somewhat subjective count of the number of instructions available in the configuration (excluding 1604-B instructions) which directly manipulate useful results as opposed to loading or unloading of block of data or results. A1 provides four such instructions; A2, 13 such instructions; A3 provides 56 such instructions plus numerous modifier fields (many of these

instructions are the result of incorporating the full set of operand type 1604-B instructions in associative modes); GAP provides five such instructions plus modifier fields. Because of its more sound architecture (i.e., the full integration of the A3 with 1604-B core) the programmer is provided a multiplicity of instructions and manipulation capability at no cost.

(11) Number of housekeeping and load/unload instructions. This indicates the number of instructions provided to manipulate the special requirements of the AM or to load or unload blocks of data to it. A1 provides 22 such instructions; A2, 34; A3, 24 plus modifier field. No significance can be gained from these statistics: as previously noted, the lack of a core-to-core move instruction in the A3 configuration has tremendous significance with respect to the effectiveness of the A3 configuration for the sea surveillance problem. This observation is more meaningful than the number of instructions between GAP and the A3 configuration.

(12) Number of tag bits. The tagbits are provided in addition to the 48 data bits of the 1604-B word size, and conditions may be specified on them for searching. A1 and A3 generalize the busy bit of the GAP to produce a true tag bit. The A2 machine has eight tag bits with some powerful processing provided for them.

As previously noted in the findings and conclusions, tag bits are extremely important. One can always reduce the word size of the A1, A3, or GAP to permit the remaining bits to be tag bits. However, when this is done, an incompatibility exists between words in AM and their representation in peripheral devices. When placing data on disc, one wants to have the full word occupied. If it is not, one has to read a portion of a word from AM (the non-tag position) and this operation is relatively cumbersome. Increased programming effort is required in this instance which is neither desirable or warranted. Providing tag bits avoids this necessity.

One use of the tag bits is to segment AM. This is useful when one wants to name or classify data, or when the programmer wants to have different classes of data in AM at the same time. The general ability of the tag bits may be emphasized by the programmer when he wants to have different classes of data in AM at the same time.

(13) Tag control of loading/search. This indicates whether or not the configuration provides for a selective setting of the tag bits when loading and for selective searching of them. The A1, A2, and A3 configurations provide full tag control, while the GAP configuration employs a busy bit which must be present for a search to be satisfied. The capability to provide tag control on a loading/search is a logical consequence of having tag bits.

(14) Source/sink for load/unload. This indicates the place from or to which data may be loaded or unloaded to the AM. All configurations, except A3, load and unload only to 1604-B core. The A3 configuratio because the AM becomes an integral part of the 1604-B core memory, may load or unload to any peripheral device or transfer data to other sections of 1604-B core.

(15) I/O load (1604-B cycles)/word transferred (in blocks or singly. This item refers to the load that is imposed on the 1604-B for each word transferred to or from associative memory in block transfer operations or in single word transfers.

(16) AM instructions per 1604-B word. This item refers to the number of commands for AM which can be contained in a 48-bit 1604-B word. The A1 and A2 configurations are controlled by EXF codes or by INT or OUT instructions. The A3 configuration uses a very powerful command code which can pack two commands per word. The GAP configuration uses a on command per word format. Although A1, A2, and A3 are more efficient than GAP in this regard, it is not clear that this is a significant factor. It is akin to the perennial debate as to whether one address, two address, or three address machines are more efficient for the programmer.

(17) Comparand and data shifting. This refers to the capability of a comparand and its mask to be shifted before a comparison, and whether or not data shifted to or from AM may be so shifted. A1 and A2 provide no shifting; A3 and GAP provide shifting. Although for the sea surveillance problem studied, no use was made in this report of this feature, it is believed essential, particularly for intelligence data processing.

(18) Mask control on transfer/search. This item refers to whether or not transfers and searches may be masked according to the programmer's option or if the mask is always active. From a programmer's viewpoint the A3 provides the most flexibility in this regard.

(19) Shift control on transfer/search. This item refers to whether and how the shifting of comparands and data may be controlled in transfers and in searches. The A3 control is useful.

(20) Resolver resettable. This item refers to whether or not the response resolver is resettable. Only A3 provides this feature and it also provides for automatic handling of resolver settings when processing vectors for I/O operations. This is potentially useful, since the programmer often desires to save the output of responders for processing at a later time.

(21) Configuration. This refers to the manner in which the AMs are interfaced with the 1604-B. A1, A2, and GAP are peripheral devices, with GAP featuring independent programming sequences via a special direct memory access channel. The A3 is a fully integrated configuration.

The GAP-DMA configuration suffers from its architecturally ambiguous position as a peripheral device. That is, the overhead in the 1604-B necessary to set up, control, coordinate, and synchronize the GAP (operating on the DMA channel) is very high. A3 eliminates most of this overhead by allowing direct manipulation of the associative features of the AM by new 1604-B instructions, and by allowing direct I/O between the AM and the peripheral devices as is necessary for the AM to show to advantage for the sea surveillance problem. A3 has also generalized the equipment of the GAP to allow more flexible and powerful programming. The AM itself is based upon GAP and is very similar to it.

(22) Cost relative to GAP. The costing of the AMs is based upon the estimates of what it would take to develop the GAP. AUERBACH cannot estimate precisely the GAP as no detailed breakdown of the GAP equipment was made available to them. What is,therefore, more meaningful than absolute costs is the ratio of the cost to that of GAP. The A1 costs 59 percent of GAP, A2 costs 62 percent of GAP, and A3 costs from 95 percent to 105 percent that of GAP. Although A3 generalizes GAP and is architecturally more attractive, its cost is not significantly different from that of GAP. With the A2, one can afford twice the storage capacity of the GAP. For sea surveillance, GAP would not have a significant advantage over A2. It is not clear that GAP would be better than a conventional system. A breakdown of costs for the four configurations is shown in Table 2-20. Section II, Volume II, Tables 2-14 to 2-20 illustrate the items used in costing a system with GAP technology. Thus, using techniques developed in these volumes, one can estimate other AMs as required.

2.2.3 Comments on the 1604-B

This paragraph serves to document some reservations AUERBACH Corporation has about the validity of data processing applications on the 1604-B computer. Our basic reservation is very simple: the 1604-B, although a fine computer of excellent design, has not been designed for data processing applications and is not representative of the best available in today's market. The Control Data 1604 is basically a scientific computer; it is in fact almost the last of the large-scale machines designed specifically for business

or scientific application. It is necessary, therefore, to enter a caveat. This report attempts to evaluate the very latest available in technology, an associative memory, in a hybrid configuration with a somewhat outdated and ill suited computer. The results obtained must be reviewed in this light for clearly the use of a more modern data processing oriented machine would have a significant effect on the goodness of a solution for the sea surveillance problem.

The following paragraphs discuss some points which lead to AUERBACH's basic reservations on the 1604-B.

2.2.3.1 Input-Output. The 1604-B input-output system is, by today's standard, both inefficient and cumbersome, specifically:

(1) The 1604-B requires two memory cycles for each word transferred in or out via the I/O system. True, this is better than the three cycles required by the 1604, but not nearly so good as the one cycle required by the 1604-A or machines like the IBM 7090.

(2) The input-output organization of the 1604-B is designed to handle I/O blocks of perspective size. Reading and writing variable length records is difficult and cumbersome.

(3) No scatter-gather facility and therefore no data skipping or suppression facility is provided in the 1604-B I/O system. Scatter-gather I/O systems are invaluable when processing large data bases of variable length files.

2.2.3.2 Interrupt. The 1604-B has at best a rudimentary interrupt system. By modern standards it is totally inadequate. It provides no separate masking or priority facilities for interrupts from peripheral devices; such facilities are invaluable when trying to maintain maximum access rates to devices such as disc files. The 1604 interrupt system requires a cumbersome program to determine what has caused an interrupt (and in some cases it is impossible to determine what has caused an interrupt). In other cases, at least in the 1604, it is possible to miss interrupting conditions (i.e., not have the interrupt hardware recognize that the condition has occurred). Although this has been corrected in the 1604-A, its status in the 1604-B is unknown.

2.2.3.3 Instructions. The instruction set of the 1604-B is not specifically suited to data processing and data manipulation operations. Specifically, there are bit handling instructions, no character handling instructions, no block transfer instructions, no variable length character string instructions, no translate instruction, and no facility for decimal-to-binary conversion and vice versa.

2.2.3.4 Peripheral Equipment. The disc file employed in the configurations studied here was chosen by RADC to be the CDC 818 disc file system. This system is an older design and cannot compete favorably with such systems as the IBM 1302 which is available today. Since a large data manipulation file processing problem is highly dependent upon the space available and the speedy access to data contained in mass storage, the use of the CDC disc file surely biases the results of this study.

SECTION III. CRITIQUE OF THE GOODYEAR AEROSPACE CORPORATION ASSOCIATIVE PROCESSOR

3.1 INTRODUCTION

The purpose of this section is to critique he Goodyear Associative Processor (GAP) from two viewpoints: hardware and software. This critique is intended to be brief and is limited to major topics; minor comments are given in Volume II. It is difficult to divorce a critique of the GAP from a critique of the 1604-B. Since the use of the 1604-B computer as the connecting device to the GAP cannot be considered as an ideal combination, some of the difficulties noted exist because of the 1604-B. This fact should be borne in mind.

3.2 HARDWARE STUDIES

This paragraph discusses the design (configuration and logic) of the Goodyear Associative Processor. The discussion is based on the GAP as defined in the Programming Manual and Goodyear Proposal to RADC (Appendix A, Volume II), and is based upon the experience of AUERBACH Corporation in applying and using the associative technology developed by Goodyear on a large file oriented processing problem. The comments presented here are purely qualitative and represent both different design philosophies and/or biases and the results of several months of analysis and study of the proper application of associative technology.

3.2.1 Direct Memory Access

The GAP and the 1604-B computer are formed into an hybrid associative processing system by using the direct memory access (DMA) channel specified by Goodyear Aerospace Corporation. This DMA channel provides for the 1604-B computer an input-output channel that is very similar to the channels which come as standard equipment on machines of the IBM 7090 class. Specifically, the DMA channel allows the input and output of information directly to 1604-B memory under control of an external mechanism. In the 7090 type machines, the external mechanism is provided by normal computer programming (i.e., using a starting address in block-size information) and then the mechanism attends to the details of transferring the data to or from memory as defined by the set up in the control registers. By providing other means for the initialization and manipulation of the control registers in such external mechanisms, great flexibility can be achieved. The Goodyear design for a DMA channel requires that the associative processor specify to the 1604-B input-output control logic the address of each word to be transferred. The address to be operated upon is determined internally in the associative processor. The design allows for the accessing of instructions and data via this mechanism. All this, in effect, provides the associative processor with two memories, the associative memory and the 1604-B memory. The DMA design then requires that instructions for the associative processor be resident in 1604-B memory where they are accessed in order by the associative processor. The net result is that as long as separate strings of program instructions exist, for the associative processor and the 1604-B, they may run independently of each other. The 1604-B uses its own memory for instruction and data storage, while the associative processor uses the 1604-B for the instruction and data storage and its own memory for data storage and search. It is apparent from the Goodyear Associative Processor design and the direct memory access channel, that the independence of the operation achieved by this configuration was, in fact, the goal desired. In the opinion of AUERBACH Corporation, however, all this was to no avail.

To understand why the above statement is true we must review the reasons for providing independent processing of subordinate tasks within a computer system. We refer to subordinate tasks because it is clearly intended that the associative processor be subordinate to the 1604-B (it has no real manipulative abilities of its own) and it is equally apparent that the design of the 1604-B computer will not allow the efficient use of more than one task at a time. Therefore, the activity of the associative processor is, in all cases, in support of the activity of the 1604-B. Subordinately and independently running mechanisms are frequently incorporated into computing systems to accommodate simple and time consuming tasks without the detailed attention of the main computing elements. In addition, subordinate elements are frequently used to accomplish major processing tasks in parallel when such parallelism can be balanced and controlled, and of course, when it is called for by the nature of the problem. In both cases, efficiency dictates that the main computing element has other functions to perform while the subordinate elements are performing their independent functions. For normal input-output buffer operations this overlap is accomplished by the job stacking or ring processing techniques. When new input is being read into a computer, results of previous processing are read out and current processing is undertaken.

In the 1604-B-GAP configuration, the conditions f r useful independence do not hold. The task for which the GAP is intended usually takes only a few microseconds to execute. Considerable time is required to set up the GAP for the execution of these operations; however, this is generally data loading and unloading time and it cannot be usefully overlapped with other operations in the 1604-B. When the 1604-B program decides a GAP operation is needed,it is generally at a point at which there is no useful parallel work to accomplish while the data is loaded into GAP, operated upon, and the results loaded back into the 1604-B. Therefore, the 1604-B, after initiating the full sequence of operations, waits for the GAP to provide results. In effect, during the load, search, and unload times, the 1604-B simply coordinates and equates. Under such conditions the direct memory access channel requires more elapsed time than would, for instance, the high speed input-output channel (channel 7) of the 1604-B, and, in addition, it costs more.

If there were adequate coordination facilities (such as interrupts) between the 1604-B and the GAP, it is still uncertain that useful overlaps could be obtained in their operation. Without major overlaps between the 1604-B and the GAP operation, no useful purpose is served by the DMA channel.

On the other hand, if the 1604-B were replaced by a much faster computer, then the GAP would take the form of the slow peripheral device. In particular, in a large multi-processing system (i.e., in terms of speed of loading and unloading) the GAP memory would be such that the direct memory access would be of considerable value. Another way of looking at it is to consider a multi-processing system where the computer would be working on problems other than those associated with the GAP. In this case, the direct memory access would be valuable. The main function of the direct memory access is to allow for loading and unloading of the GAP at the speed of the read-write operation of the core memories in the GAP. A reasonable recommendation for improvements in the GAP would be to consider utilizing a faster switch core,thereby reducing the time it takes to load and unload the GAP. A very important consideration in terms of recommendations would be a modification of the organizations such that the GAP could be loaded and unloaded via peripheral equipments.

3.2.2 General Comments

In light of the proceding paragraph it is still legitimate to ask what comments can be made of the efficacy of the configuration using a DMA channel. It seems to us that the GAP configuration suffers from being assembled with ad hoc features being appended to the design. There seems to be no edifying thought given to coordinated and integrated design. This is perhaps understandable if one considers that RADC is purchasing from Goodyear a technology rather than a design sophistication. For the design philosophy employed in the GAP, however, certain weaknesses are readily apparent; these are:

(1) The interrupt generating mechanism in the GAP is not sufficient to provide the proper coordination between the 1604-B and the GAP. We would rate this a design fault since there appears to be no logical reason for this inaccuracy. There is room in the coding of instructions to allow for interrupt control.

(2) The instruction repertoire of the GAP is not nearly as efficient as it might be and considerable efficiency could be gained in the overall system by a careful analysis of the uses and modes of operation of the GAP and the provision of instructions to accommodate these operations.

(3) The busy bit of the GAP represents a form of associative capability which has been provided in highly restricted form for no apparent reason. There is reason enough for the capability but none for the restriction. The busy bit might as well be a generalized tag bit as in the AUERBACH A3 configuration.

(4) The response vectors in the GAP (S, D, and E registers) are also overly restricted for no apparent reason. Again the A3 configuration illustrates how these facilities might be generalized.

(5) The design choices leading to the provision of pre-response vectors, each employing 2,000 flip-flops and a single restricted tag bit, are open to serious question. Other techniques are available for the storage of response vectors, and additional tag bits are probably of great value. The AUERBACH A2 and A3 configurations illustrate some of the choices available.

(6) The requirement for loading and unloading the GAP from an external memory, rather than from either another peripheral device and/or from an external memory, imposes a severe programming restriction on time and data handling.

These comments may be expanded upon in order to be constructive criticisms for hardware. This is accomplished regarding the instruction repertoire discussed in Paragraph 3.3. Regarding the third comment on busy bits, it should be pointed out that the use of the busy bit is to perform isolated searches on the memory while retaining the original information. Since the busy bit is stored in the core memory, it is necessary to read and write in a word position in order to change the busy bit. The alternative might be to provide over 2,000 flip-flops to perform the function of the busy bit. As an alternative it seems that the use of the core memory for the storage of these bits is less costly than the flip-flops and may be utilized by the programmer if he wishes. A recommendation in this area would be for additions in the instruction repertoire which would ignore the busy bit. The comment on response vectors being overly restricted is somewhat subjective if the programmer properly utilizes the P, V, and T indicator tags.

3.3 SOFTWARE

3.3.1 Introduction

The software critique is based upon observations made in programming the solution of the defined sea surveillance problem. It should be clear that the following comments regarding the programming of the GAP are neither complete nor universal. In other words, a programming defiency noted in this study may not exist in another application and conversely.

From a programming viewpoint, the items to be considered concern:

(1) Learning experience.

(2) Input-output of data and the LDR instruction.

(3) Multi-programming of GAP and 1604-B.

(4) Data dependent processing.

(5) Fixed field format processing.

(6) Instruction repertoire.

The latter part of this paragraph recommends a software technique (i.e., the Tag memory) devised to overcome some deficiencies noted.

3.3.2 Learning Experience

The AUERBACH personnel who were directly involved in programming this sea surveillance system for the GAP have an average of more than eight years' experience (from five to fourteen years) and they may be considered as senior programmers. Without exception, it has been our experience that it is very difficult to become familiar with and to understand the GAP. At this point in time (nine months after this study's beginning), these people are just beginning to feel that they can program GAP effectively. Thus, in comparison with other computers, it is felt that the learning curve of the GAP has the minimum slope. In other words, our experience has been that it is more difficult to learn how to program GAP than any other computer that the group has encountered. Interestingly enough, this includes the SOLOMON series of processors. To clarify this comment further, it is necessary to define what is required to "become familiar and understand GAP." Additionally, it is necessary to indicate why the GAP is difficult to understand.

It is believed that a programmer understands and is familiar with a processor when he no longer finds it necessary to consult the programming manual to accomplish normal processing. Of course it may be necessary to reference the manual when it is desired to optimize some section of programming for time and/or space considerations. Thus, it is felt that a programmer who understands GAP knows the complete instruction repertoire, the effect of instructions upon registers, and the interface dependent constraints. Usually, the programmer learns about these topics from the programming manual. This naturally leads into one of the reasons that the GAP is difficult to understand - the programming manual as presently written (Appendix A, Volume II) is lacking,and is not clear in,many details.

The programming manual requires the following additions:

(1) A general explanation of the 1604-B-GAP hybrid configuration with a diagram showing the:

- Addressable Registers
- Data Flow
- Control Lines
- Interface showing GAP "autonomy," and the other components given in the explanation.
- An explanation of all registers — both addressable and not addressable (provided that the latter may affect an instruction or its results).
- Storage of instructions and data.
    - Between 1604-B core and GAP memories.
    - Between 1604-B core and GAP registers.
- Input-output characteristics of the system detailing restrictions such as saturation's impact on GAP processing.

(2) Instructions

- Contents of all affected registers before, during, and after an instruction is executed.
- Operation sequence related to diagram given in introduction description.
- Timing on a detailed level.
- Sequences required to start and continue GAP operations.

Perhaps the most important item required in the current programming manual is a general explanation of the system. At the present time, some concepts of the system may be inferred from the manual after numerous readings, but they may not be valid. For example, "buffer" as used in the manual has no relationship to a programmer's usual (and almost universal) concept of "buffer." This example provides another reason why the GAP is difficult to understand.

The GAP instruction repertoire — data processing tools — is in the main a completely new set of operations; that is, there exists little similarity between the GAP and other processors' instruction sets. Thus, a programmer experienced in using instructions of conventional processors, where all instruction repertoires are somewhat similar, is in the GAP sense, an inexperienced programmer. Indeed, the conventional experienced programmer is somewhat handicapped in at least the following two ways:

(1) There exists some GAP instructions that are similar to "conventional" instructions; thus, there exists some rationale for using conventional processing techniques.

(2) Any rationale for using conventional processing techniques for GAP processing biases a solution incorrectly.

The situation is somewhat analagous to replacing a tradesman's hand tools with power tools.

The difficulties noted in learning the GAP, from a programmer's viewpoint, can be greatly obviated with a resulting decrease in required learning time by providing a comprehensive pedagogically sound programming manual.

3.3.3 Input-Output of Data and the LDR Instruction

There are two types of data inputs: associative memory storage inputs and inputs to addressable registers. In addition, certain instructions (erase, decrement and increment index, exact match) permit an actual modification (or simulation) of data which may in some sense be considered as data input. Data outputs include transferring associative memory words contents, addresses, or numerical count into the 1604-B core memory. The purpose of this paragraph is to point out some difficulty experienced in using them.

Consider the LDR instruction. This instruction specifies:

Bits	Contents
36-41	S, Shift Count
24-35	R, an Associative Memory Address
11- 0	N, a Word Count

where R is the beginning address. The LDR instruction is a supplementary instruction for 18 other instructions, including search instructions.* The function ** of the LDR is to:

(1) Specify the number of places the comparand is to be shifted. In input-output operations, the comparand is used as a one word buffer between the GAP and the 1604-B core memory. In search operations, it specifies the number of places that the comparand is shifted prior to the search operation.

(2) The R and N fields segment the associative memory into R+N-1) contiguous words beginning at location R. The ruction is performed using the data in this segment.

The main purpose of this paragraph is to consider the difficulty in forming this instruction. If S, R, and N are known before run time (i.e., their values are fixed and constant), then no difficulty occurs. But suppose that processing is data dependent (i.e., the number of words to be unloaded, for example, are not known, nor is the starting address), then the programmer formulates LDR by the following algorithm (or slight variations).

(1) Read address of desired "first" responder, which is R. The LDR instruction is not applicable; therefore,

* This paragraph also applies to the LDR instruction used for search instructions.
**One or more of S, R, and N may not be applicable.

this address is written into 1604 memory location H_1 in bits 35 through 24; all other bits are zero.

(2) Read count of responders which is written into location H_2 in bits 11 through 0, all other bits zero filled.

(3) Halt the GAP processing.

(4) Using 1604-B instructions, formulate LDR by merging H_1 and H_2.

(5) Restart the GAP processing.

This sequence of operations occurred more than 80 percent of the time since the R and N fields were often data-dependent.

A solution to this problem is perhaps not to zero fill when reading out addresses and counts of responders. Then H_1 and H_2 may be equal and the address of the next LDR instruction.

It was often the case that the LDR field values remained constant (after being determined) for long sequences of instructions. It became tiresome repeating this instruction to the extent that a programmer started to apply the instruction or specify some field incorrectly,for operations not requiring the instruction or field.

The use of the LDR instruction as applied to search instructions is not clear. It may be used, by implication, to divide the memory into segments of different data classes. But if this is true, what happens to respective portions of the D, E, and RS buffers not in the segmented portion? What happens if a buffer advance is given: does R advance by 1 also? What happens if R and N apply to the lower memory and the P bit specifies the upper memory? etc. On the other hand, these questions seem to be so unreasonable (but valid, if all fields of LDR are specified for a search instruction) to assume that only the S field of the LDR instruction is applicable in search operations. But it was often required to partition the memory into segments with a particular set of instructions applicable to particular memory segments. For example, the associative memory may contain data that is not relative to the current subroutine being performed. Therefore, it is not desired to perform operations of the current subroutine on irrelevant data (i.e., data to be used by subsequent subroutines). This associative memory segmentation required for any GAP instruction may be accomplished by hardware or by software techniques. One way to accomplish segmentation by hardware is to extend the LDR instruction to make it applicable to all GAP instructions, if desired. This study accomplished the desired segmentation by software, via the subroutine SPO (Volume II, Appendix C).

3.3.4 Multi-Programming of GAP-1604-B

From the GAP viewpoint, it is an autonomous unit performing instructions after it starts operating until a Halt instruction occurs. The GAP is started by a 1604-B program containing either a resume or a force instruction. The 1604-B may also halt the GAP by a clear operation. Thus, the GAP may be "turned on or off" by a 1604-B routine; the GAP may turn itself off, but not on. While GAP is operating, the DMA permits the 1604-B to simultaneously execute a sequence of instructions.

Considering this configuration from a systems viewpoint, it has often been found that a sequence of 1604-B instructions (LDR instruction formulation or arithmetic operations) were required in the mainstream of GAP processing. It is then required to interleave these 1604-B instructions into the set (if any) of currently operating 1604-B instructions. For example, suppose that this point has been reached. The GAP halts and the main 1604-B instruction set continue. It can sense whether or not GAP is in operation; however, several questions occur:

(1) When does the mainstream 1604-B instruction set sense GAP activity? Every k microseconds? Every k instructions? After the kth subroutine?

(2) If the GAP is inactive, does it mean that it is now free ("Free" is not synonymous with "inactive")? Or does it mean that it requires some 1604-B instruction sequence?

(3) If it (GAP) does need some 1604-B instruction set, to what 1604-B location is control transferred by the mainstream 1604-B program?

(4) After the 1604-B set of instructions required by the GAP are executed, to what 1604-B address is control transferred to continue GAP processing. Similarly, how is the mainstream 1604-B program resumed?

Before attempting to answer this question, consider another example. Suppose that data being processed by GAP had a higher priority than 1604-B processing and a penalty (for example, lost radar data) was paid for delay of GAP processing. If the data being processed by GAP is required from a peripheral device, it must be read by a 1604-B instruction into the 1604-B before it can be transferred into GAP by a GAP instruction. In this example, a programmer should execute his 1604-B input instruction k microseconds before he needs the data, where k is equal to the sum of the time to read the peripheral device data into 1604-B core memory (in this time period the GAP may resume) and the time to transfer the data from core memory to GAP (which the GAP executes with a proper LDR).

Questions (3) and (4) may be answered by use of the Store Program Counter and Jump instruction to a Halt instruction. If the GAP needs some intermediate 1604-B instructions, the stored program counter indicates where these instructions are performed and a resume instruction picks up the program counter set by the Halt instruction. If GAP is finished processing, it does not perform the Store Program Counter and Jump Instruction; therefore, the counter, which is not set, may be used as a flag. This device then answers question (2) on whether the GAP is free or waiting for some intermediate set of 1604-B instructions.

The preceding technique, however, does not answer question (1). The current design of the system dictates that either:

(1) All 1604-B mainstream routines be shorter than GAP processing if the GAP cannot be delayed, or

(2) The 1604-B mainstream routines periodically sense the GAP to see if it is active. If not active, the test of the stored program counter determines the status of the GAP (i.e., waiting for intermediate processing or available for further use).

The first alternative may be difficult to control. If it (time for 1604-B processing) cannot be controlled, data may be lost in the latter example. A suggested solution to answer question (1) is to provide an interrupt capability for GAP inactive.

3.3.5 Data Dependent Processing

On several occasions, programming difficulty in data dependent GAP processing was commented upon in the preceding paragraphs; for example, the awkwardness of setting the LDR fields given in the algorithm of Paragraph 3.3.3. The purpose of this paragraph is to extend these comments to other instructions.

Data dependent processing is where the transfer of control to one segment of processing instead of another segment is determined by the status of the data. Thus, data dependent processing requires a test for a particular data condition and a corresponding control (or transfer) operation. The test may immediately precede the control operation.

In this case, the two operations (test and control) are termed a branch operation. In addition, the control (transfer) operation which is set by the data status test may follow some intermediate (between the test and control operation) processing which usually alters the original data (see Paragraph 7.8). In this case, the control operation is usually termed a variable connector set by the test operation.

Perhaps the main feature of GAP is its ability to test the status of many data words at one time. Such a status test may be either a single test or may be a series (or logical combination) of related but individual tests. The results of such a test (and tests) are stored in the D and E buffers. Based on these buffers, certain operations may be performed. One philosophy of the GAP is that whenever a new test (note definition of "test" just given) is required, the previous test results are no longer needed and are consequently discarded.

Perhaps the weakest point of GAP is its inability to manipulate the response stores of different tests — they cannot be stored directly. This is true in regard to data dependent operations. The philosophy of destroying previous status tests results in the situation that the GAP cannot process "nested" status tests without destroying results of preceding status tests. This nested status test capability does exist on less than a word basis by use of a complex search instruction. Even in this case, the intermediate results of a complex search are lost but may be easily formed with a new test. In both the word search schemes and the complex word search instruction, a conditional test must always start at the beginning of the sequence to be tested.

In using GAP, analysis for status tests or conditional statements must be performed to reduce nested "If" statements to serial "If" statements whenever possible. If this cannot be accomplished, since intermediate results cannot be directly stored, the entire sequence of nested tests must be performed repeatedly. (Every iteration erases "first" responders.) This also yields another way out of this problem. The phrase "since intermediate results cannot be directly stored" suggests searching for a method to store intermediate results. This is given subsequently for GAP. (It is built-in hardware in the A2.)

To this point, consideration has been given to status tests of data. Consider at this time the control functions. The only way in GAP to alter a processing path based on a conditional relationship is:

(1) Jump on no responders.

(2) Jump on index high.

(3) Jump on index low.

The preceding paragraph parenthetically mentioned erasing the first responder in each iteration of the nested conditionals. Then use of the first conditional transfer instruction is clear. By definition, it is clear how the second set is used. But they may also be used in conditional data processing. For example, Jump to location C if one responder is greater than or equal to another responder. The instruction may be used if the responder's contents are less than 15 bits long, or if the data can be functionally related to its GAP address. But to use this instruction requires a transfer of address to the 1604-B, a 1604-B shift instruction on the address, and load the address into the index. Similarly, the other test argument must be entered into the instruction.

The main point being made in the preceding comments is that an index register is often required to be set to either a datum address or to an actual datum value. In the case of addresses, both read address instructions (RAF and RDA) write the address, a 1604-B core memory word,in bit positions 35 through 24; whereas the 15 rightmost bits are read into the index register. Thus, a 1604-B instruction to shift the address is required.

3.3.6 Fixed Field Processing

In many applications, data occurs in a fixed field format; that is, the value for a particular class of data always appears in some relative field position of a larger record. In this application, for example, a query, or request for information, is represented internally in a fixed field form where each "record" corresponds to a functional operation related to the query, and each record field contains some parameters for the operation. Each record is of the same format. In this type of data where an addressing relationship or topology exists,it was often required to read out the contents, or the address, of the k^{th} word (maximum k was 9) positionally related to a responder. In like manner, write operations, further test operations, erase instructions, etc., were similarly required. The situation is similar to advancing and "retarding" the buffer by $k \geq 1$ positions. Another way to view this is by setting an index register equal to k for an index register which can be applied to GAP addresses — of course, none exists.

Another comment related to fixed field processing is that the mechanism of buffer advance does not permit variable length fields unless it is known, a priori, that no more than one responder may occur. The field must be equal to the number of 48-bit words required to express the maximum word sized datum. If not, it is possible, and probable, to advance buffers into adjacent fields and still meet the search criteria.

3.3.7 Instruction Repertoire

The comments presented in preceding paragraphs are also applicable to other instructions.

Consider initially the search instructions. By a large majority, the data sizes in this study (number of bits) are equal to or exceed 48 bits. And for this reason the complex search instruction was rarely used. In fact, it was removed from two sections in which it occurred for these reasons:

(1) The analysis required to use the instruction is greater than the analysis required for data given in word sizes.

(2) It is easier to "simulate" the instruction by masks and shifts thereby building in more flexibility.

(3) It is difficult to change the "instruction" (a series of instructions) since it is interdependent.

Because the greatest majority of data used in this application was equal to or greater than 48 bits, little use was made of the shift features of the LDR.

Several instructions read out the address of responders, the count of responders, and the address of the program counter. In every instance, the balance of the word containing this information is zero filled. This probably occurs since the comparand is used as a one-word buffer in this operation. From a programming viewpoint, these instructions would be much more powerful if they "inserted" the respective data into the word in 1604-B core and did not zero fill. In many cases (e.g., formulation of subsequent LDR instructions, transfer of control operations, etc.), this feature would obviate the need for performing intermediate 1604-B operations.

It was found that the use of the busy bits could be combined with the response store registers, provided that these registers could be initialized. This fact led to an interesting and powerful supposition: that the response store elements could be individually addressed, accessed, and written into by use of the current busy bit instructions. This feature would erase a responder but not "remove" the datum from consideration.

3.4 TAG MEMORY

The motivation of the Tag Memory is the fact that GAP provides no direct methods to save and manipulate response store "vectors" of more than one test. An initial approach in this respect yielded the observation that there existed two response stores and the last one could be saved by using the E register of the upper memory as the storage device. It is then possible to manipulate the D and E registers by use of MMC, EMC, and MAX instructions to reset, set, copy, and perform boolean operations. Of course, this resulted in the loss of storage of the upper memory. This difficulty gave genesis to what is now termed the Tag Memory.

The major details of the Tag Memory are:

(1) Data is stored only in the lower memory.

(2) Tags are stored only in the upper memory.

(3) An address relationship, modulo 1024, exists between the memories by use of the P bit. Thus, the kth word in the lower memory is associated with its tag, the kth word in the upper memory. This is GAP memory word (1023 + k).

A tag has two elements, an address portion of 10 bits and a status portion of 30 bits. The address portion contains an address of a datum to which the associated datum belongs; for example, an operation's parameter in the qth word of an operation record is related to the operation indicated in the first word. Then, the address portion of the parameter's tag (1023 + q) will contain the address of the operation. This device is, programmingwise, equivalent to buffer advance.

The status portion of the tag word is used to reflect the status of the element associated with it, not the status of the element given in its address portion. The programmer must decide, <u>a priori</u>, what bit (or bits) are allocated to each status and what the setting of the bits means. Suppose for example that the tenth status bit is allocated to some value. If the datum is greater than or equal to this value, the tenth bit is set; otherwise, it is reset. This may be accomplished as follows:

(1) The lower memory data is tested.

(2) The D register is transferred to the E register.

(3) A full word of ones is written into a responder through a mask with a set bit in only that position corresponding to the tenth status bit's position.

(4) Complement the E buffer, write a full word of zeros through the same mask, erase responders, and complement the E buffer, if required.

It is therefore feasible to save at least 38 previous response store vectors by use of the Tag Memory by appropriate manipulation of the mask register (and/or the comparand). Since addresses of responders in the lower memory may be stored in the upper memory, it is also possible to query the range of addresses for responders as well.

The Tag Memory is the best determined method of using the GAP for data dependent processing. It is recommended that it be used whenever such processing occurs. However, such use is at an expense of losing half the memory's storage capability.

BIBLIOGRAPHY

Ahrons, R.W. "Superconductive Associative Memories", RCA Review 24(3):325-54, September, 1963.

Ahrons, R.W. and Burns, L. L., Jr. "Superconductive Memories" Computer Design, Vol. 1, pp. 12-19, January 1964.

Air Force Systems Command. Print-Reader Spelling Correction Shift Register (SRC). Exhibit RADC 5122. Griffiss AFB, N. Y. : Rome Air Development Center, 9 pp. and 1 p. Amendment, 1 March 1963.

Armed Services Technical Information Agency. Memory Systems: A Report Bibliography (U). ARB 10993. Arlington, ASTIA 16 pp., July 1962, (CONFIDENTIAL report).

Armed Services Technical Information Agency. Computer Decision Processes: A Report Bibliography. ARB 11 083. Arlington, ASTIA, 57 pp., July 1962.

Armed Services Technical Information Agency. Comput[illegible] and Data Processing Systems. A Report Bibliography. AD 291850. Compiled by Rulon L. Thayne, Elizabethe H. Hall, and Herman Miles., Arlington, Va. Department of Defense, ASTIA, 463 pp., August 1962.

Army Ordinance Arsenal. Multisystem Test Equipment, Frankford Arsenal Tasks.

Asher, M. "G.E. Cryogenic Associative Memory Circuit Developed", Electronic News, 19 March 1962.

ASTIA. Documentation, an ASTIA Report Bibliography. AD 267000. Arlington, ASTIA, 558 refs., 1 December 1961.

ASTIA. Automation of ASTIA-1960. AD 247000. Arlington, ASTIA, 26 pp., December 1960.

Atkin, J. and Marple, N.B. "Information Processing by Data Interrogation", IRE Trans. on Electr. Comp., Vol. EC-11, No. 2, pp. 181-187, April 1962.

Ball, J.R., Bollinger, R.C.; Jeeves, T [illegible]., McReynolds, R.C.; and Shaffer, D.H. "On the Use of the SOLOMON Parallel-Processing Computer", Proc. Fall Joint Computer Conf. Phila., Pa. (AFIPS No. 22), pp. 137-146, December 1962.

Baker, F.B. "Information Retrieval Based on Latent Class Analysis." Journal of ACM, Vol. 9 No. 4, pp. 512-521, October 1962.

Bar-Hillel, Yehoshua. "Theoretical Aspects of the Mechanization of Literature Searching". Digital Information Processors, Walter Hoffmann, ed., New York, John Wiley & Sons, Inc., pp. 406-43, 1962.

Barnett, M. P. A Hypothetical Machine for Syntax Tests, Technical Note 18. MIT, Cooperative Computing Lab., 1962.

Barton, R.S. "A New Approach to the Functional Design of a Digital Computer". Proc. W.J.C.C., pp. 393-96, May 1961.

Beesley, J.P.; Leiner, A.L.; and Rochester, N. "Design of a Large-Scale Cryogenic Memory System".

Blaauw, G.A. Multisystem Operation of the IBM System/360. IBM Corporation, Poughkeepsie, New York.

Bloom, L. A Tree Structure System for Sorting, Search and Maintenance. National Meeting of ACM, 1963.

Bloom, Leon, et al. "Card Random Access Memory (CRAM): Functions and Use". Proc. E.J.C.C. Vol. 20, pp. 147-57, New York, Macmillan, 1961.

Bloom. L.; Cohen, M.; and Porter, S. Considerations in the Design of a Computer with High Logic-to-Memory Speed Ration, presented at the AIEE Winter Meeting, Gigacycle Computer System Session, January, 1962.

Bloom, L.; Cohen, M.; and Porter, S. A Tree Structure System for Sorting, Search, and Maintenance. 1963 National Meeting of ACM.

Bourne, C.P., Bibliography on the Mechanization of Information Retrieval. Menlo Park, Stanford Research Institute, 1958.

Brooks, F.P., Jr. "Advanced Computer Organization". IFIP 1962 Proc.

Brooks, F.P. "Advanced Computer Organization-Addressing", Proc. IFIP Congress 62, Munich, Germany, pp. 564-565, 27 August- 1 September 1962.

Brown, J.R., Jr. "A Semi-Permanent Magnetic Associative Memory and Code Converter". Proc. 1961 Spec. Techn. Conf. on Nonlinear Magnetics, pp. 201-211, November 1961.

Burns, L.L.; Christiansen, D.A.; and Gange, R.A. "A Large Capacity Crypelectric Memory with Cavity Sensing", Proc. Fall Joint Computer Conf., Las Vegas, Nev. (AFIPS No. 24), pp. 91-99, November 1963.

Burke, R.G. and McCoy, R.E., Elint Operational Analysis Study (U). RADC TR-59-212. Contract AF 30(602)-1815. AD-314. Los Angeles, Electronic Control Systems, Inc., 80 pp. 31 December. (SECRET report).

Burns, L.L.; Alphonse, G.A.; and Leck, G.W. "Coincident Current Superconductive Memory".

Burns, LJCC 1963. Collection of notes on associative memories, prepared by Goodyear Aircraft Corp., Akron, Ohio, Rept No. GER 10857, October 1962.

Capobinaco, J.A. (Hughes A/C, Fullerton, Calif.); McAteer, J.E. (Hughes); Koppel, R.L. (N. American Aviation, Anaheim, Calif.). Associative Memory System Implementation and Characteristics, presented at 1964 Fall Joint Computer Conference, San Francisco, California, 27-29, October 1964.

Carter, W. C. "Mathematical Analysis of Merge-Sorting Techniques" Preprints of Proc. IFIP Congress 62. Amsterdam, North Holland Publ. Co., pp. 13-16, 1962.

Carter, W. C. "System Operation Factors". IBM Lightning Project - 5th Progress Report. Sect. 4.2.5, pp. 63-74.

Casey, R. Permanent Storage Requirements for the Position Sorter Employing the Data Interrogation Process (U). RADC TR 62-378. AD-334 328. New York, Elec. Res. Labs., Columbia Univ., 51 pp. 20 September 1961 (SECRET report).

Chadurjian, F. Comparator, U. S. Patent No. 2,973,508 February, 1961.

Cheydleur, Benjamin F. "SHIEF: A Realizable Form of Associative Memory", American Documentation. XIV:1 pp. 56-67, January 1963.

Cheydleur, B. F. Dimension: an Associative Memory. Willow Grove, Penna., Philco Computer Division, 4? pp. 20 ref., 3 December 1962.

Chu, Yaohan. "Application of Content-Addressed Memory for Dynamic Storage Allocation". RCA Review, pp. 140-152, March 1965.

Cline, R. E. Rep. 3681-10-L. AD-250696. Interim Techn. Report 1 May to 30 November 1960. Ann Arbor, Univ. of Michigan, Inst. of Science & Techn., 38 pp. December 1960.

Collection of notes on associative memories, prepared by Goodyear Aircraft Corp., Akron, Ohio. Rept No. GER 10857, October 1962.

Computer Command and Control Co., Summary of Investigation on Associative Memories. Computer Command and Control Co., Philadelphia, Pa. Rept. No. 5-101-5, January 1964.

Control Data Corporation. 10 Specifications for the 1604-B. CDC pub. No. 60110100, July 1964.

Conway, M. E. "A Multiprocessor System Design", Proc. Fall Joint Computer Conf., Vol. 24, pp. 139-146; November 1963.

Corbell, R. C. A Tunnel Diode Associative Memory, M. S. Thesis, UCLA, June 1962.

Corneretto, A. "Associative Memories", Electronic Design, Vol. 11

Corneretto, A. "Associative Memories. A Many Pronged Design Effort". Electronic Design, February 1963.

Corneretto, A. "Three-Bit Associative Memory Works at Room Temperature", Electronic Design, 5 July, 1962.

Craft, J. L.; Goldman, E. H.; and Strohm, W. B. "A Table Look-Up Machine for Processing Natural Languages", IBM J. Research & Dev. 5(3): 192-203, July 1961.

Cros, R. C. "Review of the Multi-List System for Real-Time Storage and Retrieval" by N. S. Prywes and H. J. Gray. Computing Reviews. IV: 3, p. 1?5, May 1963.

Crowe, J.W. "Trapped-Flux Superconduction Memory", IBM J. Research & Dev. 1(4):294-303, October 1957.

Danylchuck, L; Perneski, A. J.; and Sagal, M.W. "Plated Wire Magnetic Film Memories", Intermag Proceedings, Washington, D.C., April 1964.

Daubak, B.J. and C.R. Warburton. File Organization for Automated Retrieval. TR 00.946. Poughkeepsie IBM Data Systems Div. 24 pp., 26 November 1963 (presented under similar title at ACM Conference, Denver, August 1963.

Davies, P. "Design for an Associative Computer". Proc. IEEE Pacific Computer Conf. (T-147):109-117, 2 March 1963.

Davies, P. "A Superconductive Associative Memory", Proc. Spring Joint Computer Conf. San Francisco, Calif. (AFIPS No. 21): pp. 79-88, May 1962.

Davies, P. Associative Processors, presented at the IEEE Symp. on Search Memory, Los Angeles, Calif. Sponsored by the West Coast Comm., Los Angeles/Orange County Chapters, IEEE Computer Group. 26 May 1964.

Deronald, C.H. and Fotheringham, J.A. "The ATLAS Computer", Datamation, Vol. 7, pp. 23-27, May 1961.

Documentation, Inc. Research on the Development of a Storage and Search Theory. AFOSR - 1837. AD 267 606. Contract AF 49(638 91). Washington, D.C.: Documentation, Inc. 15 pp., November 1961.

Douglas, A.S. "Techniques for the Recording of, and Reference to Data in a Computer", Computer Journ. II, pp. 1-9, April 1959.

Eckert, J.P., Jr. "A Survey of Digital Computer Memory Systems". Proc. IRE XLI. pp. 1393-1406, 10 October 1953.

Electronic Industries 23, pp. 86-87, 8 August 1964.

Elec. Research Labs. Advanced Radar Resolution Techniques, Research on Ultrasonic Delay Lines. RADC TDR-62-216, Vol. I AD 285222. Contract AF30(602)-1971. Final Report F/178 (F/152). New York, Columbia Univ. 1962.

Estrin, G. "Organization of Computer Systems -- The Fixed Plus Variable Structure Computer", Proc. Western Joint Computer Conf. (NJCC No. 17):33-7, May 1960.

Estrin, G. and Fuller, R. Algorithms for Content-Addressable Memory Organization, Proc. IEEE Pacific Computer Conf. Pasadena, California (T-147):118-30, March 1963.

Estrin, G. "Organization of Computer Systems - The Fixed Plus Variable Structure Computer", Proc. WJCC, May 1960.

Estrin, G. and Fuller, R. "Algorithms for Content-Addressable Memories," Proc. 1963 Pacific Comp. Conf. IEE Spec. Publ. T-147. New York, pp. 118-30, IEEE, 1963.

Estrin, G. and Fuller, R. "Some Applications for Content-Addressable Memories", AFIPS Conf. Proc. XXIV, pp. 595-508, November 1963.

Estrin, G. and Fuller, R. "Some Applications for Content-Addressable Memories", Proc. Fall Joint Computer Conference, Las Vegas, November 1963.

Estrin, G. and Fuller, R. "Algorithms for Content-Addressable Memories", Proceedings of the Pacific Computer Conference, November 1963.

Estrin, G. and Fuller, R. "Algorithms for Content-Addressable Memories", Proc. 1963 Pacific Comp. Conf. IEEE Spec. Publ. T-147, pp. 118-30, New York, IEEE, 1963.

Estrin, G. and Fuller, R. H. "Some Applications for Content-Addressable Memories", AFIPS Conf. Proc. XXIV, pp. 495-508, November 1963.

Evans, G. R. Research on Various Phenomena for the Performance of Computer Functions: An Annotated Bibliography of Report Literature. SB 63 2, AD-405 748, Contract AP33(657)-8777. Sunnyvale, Lockheed Aircraft.

Ewing, Richard D. and Davies, Paul M. An Associative Processor, presented at 1964 Fall Joint Computer Conference. San Francisco, California, (Abacus, Inc., Santa Monica, Calif.) 27-29 October 1964.

Evreinov, E. V., and Kosarev, Yu. G. Large Scale Computing Systems of the Future. Izv. AN SSSR. Tekhn. Kibernetika 4, pp. 3-25, 1963.

Ewing, R. G. and Davies, P. M. "An Associative Processor", presented at 1964 Fall Joint Computer Conference, San Francisco, California, 27-29, October 1964. (Abacus Incorporated, Santa Monica, Calif.).

Falkoff, A. D. "Algorithms for Parallel Search Memories", Journal ACM. IX:4 pp. 488-511, October 1962.

Farrar, James M., Jr. Associative Memory Applications in Intelligence Data Processing. Federal Systems Division, IBM Corporation, Rockville, Maryland, 29 December 1961.

Feldman, J. A. Aspects of Associative Processing, MIT, Lincoln Laboratory.

Flores, Ivan. "Analysis of Internal Computer Sorting", J. Assoc. for Comp. Mach. VIII. pp. 41-80, 1January 1961.

Flores, Ivan "Computer Time for Address Calculation Sorting", J. Ass. Comp. Mach. VII 4, pp. 389-409, October 1960.

Fotheringham, J. A. et. al. "The Atlas Computer", Datamation. Vol. 7, pp. 23-27, May 1961.

Frank, R. M. and Larasus, R. B. "A High-Speed Sorting Procedure", Comm. of the ACM. III:1 pp. 20-22, January 1960.

Frie, E. H. and Goldberg, J. "A Method for Resolving Multiple Responses in a Parallel Search File", IRE Trans. on Electronic Computers, EC-10, No. 4, pp. 718-723, December 1961. (Also in Multiple Instantaneous Response File by Goldberg, et al., pp. 125-35.)

Freidkin, E. "Trie Memory", Report No. 734, prepared by Bolt, Beranek & Newman, Inc. ACM Comm. III: 9, pp. 490-499, September 1960.

French, W. L. "Associative Memory", U.S. Patent No. 3, 131-291, April 1964.

Friend, Edward H. "Sorting on Electronic Computer Systems", J. Assoc. Comp. Mach. III: 3, pp. 134-68, July 1956.

Fuller, R. H. "Content-Addressable Memory Systems", University of California, Los Angeles. Rept. No. 63-25. Contract NOnr-233(52), June 1963.

Fuller, R. H.; Tu, J. C.; and Bird, R. N. "A Woven Plated-Wire Associated Memory", Proceedings, 17th Annual National Aerospace Electronics Conf. Dayton, Ohio, May 1965.

Futami, K. et al. "The Plated-Woven Wire Memory Matrix," Intermag Proceedings, Washington, D.C., April 1964.

Gall, R. G. A Hardware Integrated General Purpose Computer/Search Memory presented at 1964 Fall Joint Computer Conference, San Francisco, California, 27-29 October 1964. (Goodyear Aerospace Corp., Akron, Ohio).

Gelerenter, H.; Hansen, J. R.; and Gerberich, C. L. "A Fortran-Compiler List-Processing Language", JACM, Vol. 17, April 1960.

Goldberg, J. "Binary Tests for Two-Terminal, Simultaneous Action, Data Retrieval Machines," Suppl. B to Quart. Rept. 2. Contract AF30(602)-2142, Menlo Park, Stanford Res. Inst., 1960.

Goldberg, J. "Review of 'A Magnetic Associative Memory' by J. R. Kiseda, et al". IEE Trans. on Electronic Computers, EC-12, No. 4, p. 416, August 1963.

Goldberg, J. and Green, M. W. Multiple Instantaneous Response File. RADC-TR-61-223. AD 266 169, contract AF30(602)-2142. Menlo Park, Stanford Research Inst., 220 pp., August 1961.

Goldberg, J. and Green, M. W. Large Files for Information Retrieval Based on Simultaneous Interrogation of all Items, presented at Symp. on Large Capacity Memory Techniques, Washington, D.C. Sponsored by U.S. Navy, Ofc. Naval Research, May 1961.

Gall, R. G. Preliminary System Search Time Analysis - NTDS Search Memory. GER 11152, Goodyear, 43 pp. May 1963.

Goodyear Aircraft Corp. Collection of Notes on Associative Memory. Report GER 10857, Akron, Ohio: Goodyear Aircraft Co., October 1962.

Goodyear Aerospace Corporation. Proposed 2048-Word Associative Memory Equipment, RADC PR 64-844, GAP-2549, 8 June 1964.

Goodyear Aerospace Corporation, Preliminary Programming Manual for the RADC 2048 Word Memory, Appendix A of Goodyear Aerospace Corporation No. AP-112286.

Goodyear Aircraft Corp. Applications of Parallel Search Memories, Proposal for Study of GAP-1933. Akron, Ohio: Goodyear Aircraft Corp., 157 pp., 19 March 1963.

Gray, H. J., Jr. and Parker, E. J. Information Retrieval and the Design of More Intelligent Machines. AD60URI, AD243 490. Contract DA 36-039-sc-75047, Philadelphia, School of EE, University of Pennsylvania, 71 pp., 3 October 1960.

Green, B. F. "Computer Languages for Symbol Manipulation", IRE Trans. Electronic Comp. EC-10(4): 729-35, December 1961.

Green, M. W. "A Cryogenic Multiple Instantaneous Response File". Contract AF39(602)-2142. Appendix C. Multiple Instantaneous Response File, by Goldberg, RADC TR-61-233. AD 266 169. Menlo Park, Stanford Research Institute, pp. 137-155, August 1961.

Griffith, J. E. "An Intrinsically Addressed Processing System", IBM Systems Journal. Volume II, pp. 182-199, September-December 1963.

Griffith, J. "Techniques for Advanced Information Processing Systems", 1st Congress on the Information Systems Sciences, (MITRE-ss 7).

Giuliano, Vincent E. Requirements of Future Computer Memories for Document Processing. Presented at the American Documentation Institute, 27th Annual Meeting, Philadelphia, Pa. (A. D. Little & Co.), 5-8 October, 1964.

Giuliano, V. E. and Jones, P. E. Linear Associative Information Retrieval, Working Memo, Acorn-o, prepared by Harvard U., August 1962.

Haynes, John L, and Minnick, Robert C. Magnetic Core Access Switches, RADC-TR-61-117B. Contract AF30(602)-2227. AD-260 118. Menlo Park, Stanford Research Institute, 264 pp., May 1961.

Haynes, M. K. Cryotron Storage, Arithmetic and Logical Circuits, presented at the Symp. on Superconductive Techniques for Computing Sys., Washington, D. C. Sponsored by the U. S. Navy, Ofc. Naval Research.

Heckler, C. H., Jr. "Magnetic Realizations for MIRF Employing One Flux Path per File Item". Multiple Instantaneous Response File by J. Goldberg, et al., pp. 195-220.

Hellerman, H. A Directory Control System for Multiprogramming, IBM, Thomas J. Watson Res. Cntr. N.Y., Rept. No. RC-1095, October 1963.

Hess, Herman. "A Comparison of Disks and Tapes", Comm. ACM VI: 10, pp. 634-38, October 1963.

Holland, J. H. "An Iterative Circuit Computer Constructed of Microelectronic Components and System", Proc. Western Joint Computer Conf., San Francisco, California. (NJCC No. 17), pp. 259-265, May 1960.

Holland, J. H. "A Universal Computer Capable of Executing an Arbitrary Number of Subprograms Simultaneously".

Hollander, Gerhard L. "Quasi-Random Access Memory System", Proc. EJCC 1956 (AIEE Spec. Publication T-92). New York, American Institute of Electrical Engineers, 1957.

Hollander, G. Introduction to Search Memories, presented at the IEEE Symp. on Search Memory, Los Angeles, Calif. Sponsored by the West Coast Comm., Los Angeles, Orange County Chapters, IEEE Computer Group.

Hollander, G. L. and Porter, S. N. Evaluation Criteria for Associative Memories, Prepared for Code RAWID Rome Air Development Center, Griffiss Air Force Base, N.Y. Contract AF30(602)-3108.

Holt, Anatol W. Some Theorizing on Memory Structure and Information Retrieval, Princeton, N.J.: Applied Data Research, Inc., 27 pp. (Edited version of "Programmed Memory Organization". ACM Nat'l Conference, 1963). 4 October 1963.

Horvath, R. Integrating the Search Memory with the USQ-20 Computer". GER-11621, Akron, Ohio, Goodyear Aerospace Corporation, 4 June 1963.

Howard, R. A.; Wells, P. E.; Cann, L; and Davis, J. S. Investigation of Woven-Screen Memory Techniques, presented at Symp. on Large Capacity Memory Techniques, Washington, D. C. Sponsored by U. S. Navy, Ofc. Naval Research, May 1961.

Hubbard, G. U. An Analysis of Merging from Random Access Storage Devices, Report 16, 01, 001, 084. San Jose: IBM ASDD, 47 pp. (Presented at 18th ACM Meeting, Denver, 1963.) 26 June 1963.

Hughes Aircraft Co. Hughes Associative Memory, Report MA62-16-29, Fullerton, California, Hughes Aircraft Co., 16 pp. March 1963.

IBM. IBM Lightning Project, Final Repts. and Quarterly Prog. Repts., No. 6(3) (Appendix B covering 1 January 1959, 31 May 1960), AD 250678.

Isaac, E. J. and Singleton, R. C. "Sorting by Address Calculation", Journ. ACM III: 3, pp. 169-174, July 1956.

Iverson, K. E. A Programming Language, New York, John Wiley and Sons, pp. 275, 1962.

Johnson, L. R. and McAndrews, M. H. "On Ordered Retrieval from an Associative Memory", IBM, J. Res. Develop. 8, 2, pp. 189-193, April 1964.

Johnston, L. R. "An Indirect Chaining Method for Addressing on Secondary Keys", Comm ACM IV 5, pp. 218-221.

Joseph, E. C. and Kaplan, Albert "Target Track Correlation with a Search Memory", Proc. 6th National MIL-E-CON, June 1962.

Kaplan, Albert. "A Search Memory Subsystem for a General Purpose Computer". AFIPS Conf. Proc. XXIV, pp. 193-200. November 1963.

Katz, Jesse H. Application of a Parallel-Search Memory. Systems and Programming Research Report No. 7, TRW Computer Division, Canoga Park, California, 10 December 1962.

Kaufman, B. and Ulzurrun, E. "A New Technique for Using Thin Magnetic Films as a Phase Script Memory Element", Proc. Fall Joint Computer Conf., Las Vegas, Nev. (AFIPS No. 24):59-65, November 1963.

Kenney, J. L. An Economical Subroutinized Sorting Technique, TR 00 1034. Poughkeepsie, N. Y., IBM Data Systems Div., 7 pp. 1 August 1963 (Presented at ACM Conference, Denver, 27-30 August 1963).

Kessel, B and DeLucia A. "A Specialized Library Index Search Computer", Proc. W. J. C. C., pp. 57-59, March 1959.

Kilburn, T.; Edwards, D. B. G.; Lanigan, N. J.; and Sumner, F. H. "One-Level Storage System", IRE Trans. on Electronic Computers, Vol. EC-11, pp. 233-235, April 1962.

King, G. "Table Lookup Procedures in Language Processing. Part 1: The raw text", IBM J. Research & Dev. 5(2):86-92, April 1961.

Kiseda, J. R.; Peterson, H. E.; Seelbach, W. C.; Teig, M. "A Magnetic Associative Memory", IBM J. of Research & Development. Vol. 2, pp. 106-121, April 1961.

Koerner, R. J. Memory Array Searching System. U. S. Patent 3,031,650, 24 April 1962.

Koerner, R., and Scarborough, A. "Theory and Organization of a Representative Search Memory", presented at the IEEE Symp. on Search Memory, Los Angeles, Calif. Sponsored by the West Coast Comm., Los Angeles/Orange County Chapters, IEEE Computer Group, 26 May 1964.

Kuttner, P. "The Rope Memory -- A Permanent Storage Device", Proc. Fall Joint Computer Conf., Las Vegas, Nev. (AFIPS No. 24):45-57, November 1963.

Lecerf, Yves, Intrinsic Machine Addressing in Automatic Translation, MTL 611283.

Ledley, R.S. Organization of Large Memory Systems, presented at the Symp. on Large Capacity Memory Techniques, Washington, D.C., sponsored by U.S. Navy, Ofc. Naval Research, May 1961, M.C. Yovits, Ed. New York, Macmillan, pp. 15-51, 1962.

Lee, C.Y. "Intercommunicating Cells, Basis for a Distributed Logic Computer", Proc. Fall Joint Computer Conf. Philadelphia, Pa. (AFIPS No. 22): 130-6.

Lee, C.Y. and Paull, M.C. "A Content-Addressable Distributed Logic Memory with Application to Information Retrieval", Proc. IEEE, LI: 6, pp. 924-32, June 1963.

Lee, E.S. "Solid State Associative Cells", Proceedings of the Pacific Computer Conference, California Institute of Technology, 15-16 March 1963.

Lee, E.S. "Semiconductor Circuits in Associative Memories", Proc. IEEE Pacific Computer Conf., 93-101, Pasadena, California (T-147) March 1963.

Lee, E.S. "Solid State Associative Cells", Proc. 1963 Pacific Computer Conf. IEEE Spec. Publ. T-147, New York, IEEE, pp. 96-108, 1963.

Lee, Edwin S. "Associative Techniques with Complementing Flip-Flop". Proc. SJCC. Vol. XXIII, pp. 381-394, Baltimore, Spartan Book Co., 1963.

Lekfovitz, D. and Prywes N.S. "Automatic Stratification of Information" Proc. SJCC. XXIII, pp. 229-40, May 1963.

Lewin, Morton H. "Retrieval of Ordered Lists from a Content-Addressed Memory". RCA Review, XXIII: 2, pp. 215-29, June 1962.

Lewin, M. H.; Bellitz, H.R.; and Rajchman, J.A. "Fixed, Associative Memory Using Evaporated Organic Diode Arrays", AFIPS Conf. Proc. XXIV, pp. 101-106, November 1963.

Librascope Division. L-3000 Series Mass Memory File. Glendale: Librascope Division, General Precision, Information Systems Group, (n.d.), 35 pp.

Lin, Andres D. "Key Addressing of Random Access Memories by Radix Transformation". Proc. SJCC.XXIII, pp. 355-366, May 1963.

Little, A.D. Automatic Message Retrieval Studies for the Design of an English Command and Control Language System, Final Report, Contract Number AF19(628)-256, November 1963.

Litton Industries. Airborne Ferret Data Processing Techniques (U), AD-323 035. Contract AF33(616)-5560. Beverly Hills, Electric Equipments Division, Litton Ind., 63 pp. 8 May 1961, (SECRET report).

Long, T.R. "Electrodeposited Memory Elements for a Nondestructive Memory". Journ. Appl. Phys. Supplement XXXI, pp. 1235-1245, 5 May 1960.

Loreto, D.R.; Lay, C.A. et al. Project Lint (U). RADC TR-59-14. AD 304-733. 61 pp. February 1959. (SECRET report).

Lucas, Joseph R. Capt. USAF. CAM Developments. Technical Memorandum R51/46, January 1964.

Lussier, Robert, R. and Schneider, Robert P. "All Magnetic Content-Addressed Memory". Electronic Industries, pp. 92-98, March 1963.

McAteer, J. The Search Memory in an Information Retrieval System, presented at the IEEE Symp. on Search Memory, Los Angeles, California. Sponsored by the West Coast Comm., Los Angeles/Orange County Chapter, IEEE Computer Group, 26 May 1964.

McAteer, J. E.; Capobianco, J. A.; and Koppel, R. L. "Associative Memory System Implementation and Characteristics". Proc. AFIPS 1964 Fall Joint Computer Conf., pp. 81-92.

McCormick, B. H. and Divilbiss, J. L. Tentative Logical Realization of a Pattern Recognition Computer". Report No. 403, prepared by U. Illinois, Digital Computer Lab. Urbana, Illinois, 1961.

McDermid, W. I. and Petersen, H. E. "A Magnetic Associative Memory System", IBM J. Res. & Dev. Vol. 1, pp. 59-62, January 1961.

McGee, W. C. "Review of 'SHIEF: A Realizable Form of Associative Memory' by B. F. Cheydleur". Computing Reviews. IV: 5, p. 227, September 1963.

MacGowan, Roger A. and Whigham, Wilton L. General Purpose Digital Computer for Engineering and Scientific Applications (Characteristics and Trends). RE TR-61, AD 249 699. Huntsville, Alabama, Army Ballistic Missile Agency, 33 pp. 16 January 1961.

Mann, Horace T. and Rogers, John "A Cryogenic 'Between Limits' Associative Memory". Proc. IRE Natl. Aerospace Electronics Conf., Dayton, Ohio, Institute of Radio Engineers, pp. 359-62, May 1962.

Marple, N. B. and Atkin, J. "Information Processing by Data Interrogation", IRE Trans. Electronic Compu. EC-11(2) 181-7, April 1962.

Magnavox Research Labs. Mathematical Models for Information Systems Design. RADC-TR-61-196. AD 266577. Torrance, Magnavox Res. Labs., 178 pp. Prepared by R. M. Hayes, 27 October 1961.

Marill, T. M. Combinatorial Aspects of Information Retrieval. Res. Rept. ECPX 0027. Contract AF30(634)-1404. AD 252 031 L. Cambridge, Bolt, Bernanek and Newman, 9 pp., November 1960.

Miller, S. W. Fundamental Investigation of Digital Computer Storage and Access Techniques. RADC-TR-61-117A. Contract AF30(602)-2277. AD-260 117. Menlo Park, Stanford Research Inst., 90 pp. May 1961.

Miller, S. W. and Haynes, J. L. "Investigation of Storage and Access Techniques Suitable for Use in Large-Capacity Digital Memories". Large-Capacity Memory Techniques for Computer Systems, pp. 1-14. Edited by M. C. Yovits. New York, Macmillan, 1962.

Minker, J. Associative Memory System Implementation and Characteristics, reviewed for Computing Reviews for March-April 1965 issue. Authors: McAteer, J. E., Capobianco, J. A., and Koppel, R. L.

Minker, J. and Shindle, W. E. An Associative Processor, reviewed for Computing Reviews for March-April 1965 issue. Authors: Ewing, Richard G., and Davies, Paul, M.

Minker, J., and Shindle, W. E. A Hardware-Integrated GPC/Search Memory, reviewed for Computing Reviews for March-April 1965 issue. Authors: Gall, Russell, G.

Minker, Jack. Data Storage and Retrieval: Disc-Based Systems, presented at the IEEE Workshop on Programming Languages for Command and Control, Plymouth, Mass., 16 , 17 September 1965.

Minnick, R. C. "Magnetic Comparators and Code Converters", Switching Theory in Space Technology, pp. 193-204. Howard Aiken and Wm. F. Main, eds. Stanford Univ. Press 1963.

Mooers, C. N. Seven System Models. RADC-TR-59-173. Zator Tech. Bull. 133, Part II, Contract AF30(602)-1900. Cambridge, Zator Co., 39 pp. August 1959.

Mooers, C. N. The Application of Simple Pattern Inclusion Selection to Large-Scale Information Retrieval Systems. RADC-TN-59-157, AD 215 434, Zator Tech. Bull. 131. Contract AF30(602)-1900. Cambridge, Zator Co., 20 pp. April 1959.

Mooers, C. N. Extensions of Pattern Inclusion Selection. RADC-TR-59-169. Zator Tech. Bull. 133, Part I. Contract AF30(602)-1900. Cambridge, Zator Co., 39 pp. August 1959.

Mooers, C. N. The Intensive Sample Test for the Objective Evaluation of the Performance of Information Retrieval Systems - RADC-TN-59-160. Zator Tech. Bull. 132. Contract AF30(602)-1900. Cambridge, Zator Co., 20 pp. August 1959.

Mooers, C. N. "Choice and Coding in Information Retrieval Systems". Trans. of IRE-PGIT, Vol. IV, (also Zator Tech. Bull. No. 100 Boston. PGIT-4 Zator Company). September 1954.

Mooers, C. N. "Zatocoding Applied to Mechanical Organization of Knowledge". American Documentation, Vol. II, No. 1, pp. 20-32, (ZBT 62a, Cambridge, Zator Company). January 1951.

Morenoff, Edw. and McLean, John B. Theory of a Multiple Tape Queuing System and Its Application to Electronic Systems. RADC-TDR-62-167, AD-276 359. Rome, N.Y. Rome Air Dev. Ctr., 15 pp. May 1962.

Newcombe, Howard B. and Kennedy, James M. "Record Linkage", Comm. ACM, VI: 11, pp. 563-66 November 1962, Review: Computing Reviews, IV: 3, 135, May 1963.

Lucas, Joseph R. Capt. USAF. CAM Developments. Technical Memorandum R51/46, January 1964.

Lussier, Robert, R. and Schneider, Robert P. "All Magnetic Content-Addressed Memory". Electronic Industries, pp. 92-98, March 1963.

McAteer, J. The Search Memory in an Information Retrieval System, presented at the IEEE Symp. on Search Memory, Los Angeles, California. Sponsored by the West Coast Comm., Los Angeles/Orange County Chapters, IEEE Computer Group, 26 May 1964.

McAteer, J.E.; Capobianco, J.A.; and Koppel, R.L. "Associative Memory System Implementation and Characteristics". Proc. AFIPS 1964 Fall Joint Computer Conf., pp. 81-92.

McCormick, B.H. and Divilbiss, J.L. Tentative Logical Realization of a Pattern Recognition Computer". Report No. 403, prepared by U. Illinois, Digital Computer Lab. Urbana, Illinois, 1961.

McDermid, W.L. and Petersen, H.E. "A Magnetic Associative Memory System", IBM J. Res. & Dev. Vol. 1, pp. 59-62, January 1961.

McGee, W.C. "Review of 'SHIEF: A Realizable Form of Associative Memory' by B.F. Cheydleur". Computing Reviews. IV: 5, p. 227, September 1963.

MacGowan, Roger A. and Whigham, Wilton L. General Purpose Digital Computer for Engineering and Scientific Applications (Characteristics and Trends). RE TR-61, AD 249 699. Huntsville, Alabama, Army Ballistic Missile Agency, 33 pp. 16 January 1961.

Mann, Horace T. and Rogers, John "A Cryogenic 'Between Limits' Associative Memory". Proc. IRE Natl. Aerospace Electronics Conf., Dayton, Ohio, Institute of Radio Engineers, pp. 359-62, May 1962.

Marple, N.B. and Atkin, J. "Information Processing by Data Interrogation", IRE Trans. Electronic Compu. EC-11(2) 181-7, April 1962.

Magnavox Research Labs. Mathematical Models for Information Systems Design. RADC-TR-61-196. AD 266577. Torrance, Magnavox Res. Labs., 178 pp. Prepared by R. M. Hayes, 27 October 1961.

Marill, T.M. Combinatorial Aspects of Information Retrieval. Res. Rept. ECPX 0027. Contract AF30(634)-1404. AD 252 031 L. Cambridge, Bolt, Bernanek and Newman, 9 pp., November 1960.

Miller, S.W. Fundamental Investigation of Digital Computer Storage and Access Techniques. RADC-TR-61-117A. Contract AF30(602)-2277. AD-260 117. Menlo Park, Stanford Research Inst., 90 pp. May 1961.

Miller, S.W. and Haynes, J.L. "Investigation of Storage and Access Techniques Suitable for Use in Large-Capacity Digital Memories". Large-Capacity Memory Techniques for Computer Systems, pp. 1-14. Edited by M.C. Yovits. New York, Macmillan, 1962.

Minker, J. Associative Memory System Implementation and Characteristics, reviewed for Computing Reviews for March-April 1965 issue. Authors: McAteer, J. E., Capobianco, J. A., and Koppel, R. L.

Minker, J. and Shindle, W. E. An Associative Processor, reviewed for Computing Reviews for March-April 1965 issue. Authors: Ewing, Richard G., and Davies, Paul, M.

Minker, J., and Shindle, W. E. A Hardware-Integrated GPC/Search Memory, reviewed for Computing Reviews for March-April 1965 issue. Authors: Gall, Russell, G.

Minker, Jack. Data Storage and Retrieval: Disc-Based Systems, presented at the IEEE Workshop on Programming Languages for Command and Control, Plymouth, Mass., 16 , 17 September 1965.

Minnick, R. C. "Magnetic Comparators and Code Converters", Switching Theory in Space Technology, pp. 193-204. Howard Aiken and Wm. F. Main, eds. Stanford Univ. Press 1963.

Mooers, C. N. Seven System Models. RADC-TR-59-173. Zator Tech. Bull. 133, Part II, Contract AF30(602)-1900. Cambridge, Zator Co., 39 pp. August 1959.

Mooers, C. N. The Application of Simple Pattern Inclusion Selection to Large-Scale Information Retrieval Systems. RADC-TN-59-157, AD 215 434, Zator Tech. Bull. 131. Contract AF30(602)-1900. Cambridge, Zator Co., 20 pp. April 1959.

Mooers, C. N. Extensions of Pattern Inclusion Selection. RADC-TR-59-169. Zator Tech. Bull. 133, Part I. Contract AF30(602)-1900. Cambridge, Zator Co., 39 pp. August 1959.

Mooers, C. N. The Intensive Sample Test for the Objective Evaluation of the Performance of Information Retrieval Systems - RADC-TN-59-160. Zator Tech. Bull. 132. Contract AF30(602)-1900. Cambridge, Zator Co., 20 pp. August 1959.

Mooers, C. N. "Choice and Coding in Information Retrieval Systems". Trans. of IRE-PGIT, Vol. IV, (also Zator Tech. Bull. No. 100 Boston. PGIT-4 Zator Company). September 1954.

Mooers, C. N. "Zatocoding Applied to Mechanical Organization of Knowledge". American Documentation, Vol. II, No. 1, pp. 20-32, (ZBT 62a, Cambridge, Zator Company). January 1951.

Morenoff, Edw. and McLean, John B. Theory of a Multiple Tape Queuing System and Its Application to Electronic Systems. RADC-TDR-62-167, AD-276 359. Rome, N.Y. Rome Air Dev. Ctr., 15 pp. May 1962.

Newcombe, Howard B. and Kennedy, James M. "Record Linkage", Comm. ACM, VI: 11, pp. 563-66 November 1962, Review: Computing Reviews, IV: 3, 135, May 1963.

Newell, A. and Tonge, F. "An Introduction to Information Processing Language-V", Communs ACM 3(4):205-11, April 1960.

Newhouse, Vernon L. Applied Superconductivity, General Electric Research Laboratory.

Newhouse, V. L. and Fruin, R. E. A Cryogenic Data Addressed Memory, Rep. 62-RL-2985E. Schenectady, General Electric Res. Lab., April 1962. Also: Electronics, pp. 31-36, 4 May, 1962.

Newhouse, V. L. and Fruin, R. E. "A Cryogenic Data Addressed Memory", Proc. SJCC. XXI, pp. 89-100, May 1962.

Nissim, Samuel. Nanophile Digital Organizations. M70-3U6. Canoga Park, TRW Computer Div., 40 pp. August 1962.

Orr, W. K. Look Ahead Logic Simplified. (To be published.)

Petersen, H. E. et al. "A Magnetic Associative Memory", IBM Journal of Research and Development, Vol. 5, April 1961.

Porter, R. W. "A Large-Capacity Document Storage and Retrieval System", Large-Capacity Memory Techniques, pp. 351-360, M. C. Yovits, ed. New York, Macmillan Co., 1962.

Porter, S. Use of Multiwrite for General Program Ability of Search Memories, presented at the IEEE Symp. on Search Memory, Los Angeles, California. Sponsored by the West Coast Comm., Los Angeles/Orange County Chapters, IEEE Computer Group, 26 May 1964.

Postley, J. A. "Contrasts in Large File Memories for Large-Scale Computers", Proc. W.J.C.C., pp. 193-94, 1959.

Pritchard, J. P. and Wald, L. D. "Design of a Fully Associative Cryogenic Data Processor", Proceeding of the Intermag Conference, April 1964.

A Proposal for the Study of Associative Processing Techniques. Report No. FP 63-16-276, Hughes Aircraft Company, 14 October 1963.

Prywes, N. S. and Gray, H. J. Multi-List Organized Associative Memory, prepared for University of Pennsylvania, Moore School of Electrical Engineering, January 1962.

Prywes, N. S. and Gray, H. J. "The Organization of a Multi-List Type Associative Memory", IEEE Trans. Communs and Electronics, CE-82, 488-92, September 1963.

Prywes, N. S. and Gray, H. J. "The Organization of a Multi-List Type Associative Memory", AIEE Special Publication S136, pp. 87-101, January 1962.

Prywes, N. S. and Gray, H. J. "The Multi-List System for Real-Time Storage and Retrieval", Preprint of Proc. IFIP Congress 62, pp. 112-116, Amsterdam, North Holland Publishing Co., 1962.

Prywes, N.S. and Landauer, W.I. An Automated Sea Surveillance System, A Report prepared by Computer Command and Control Co. Rept. No. 4-101-4. January 1964.

Rajchman, J.A. "Computer Memories -"A Survey of the State-of-the-Art", Proc. IRE. Vol. 49, No. 1, pp. 104-27, January 1961.

Rajchman, J.A. "Magnetic Memories: Capabilities and Limitations", Computer Design. II:8, pp. 32-37, September 1963.

Raphael, Bertram. SIR: A Computer Program for Semantic Information Retrieval, Submitted to the Department of Mathematics on April 8, 1964, in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Reitz, G. Language Data Processing with Search Memories, presented at the IEEE Symp. on Search Memory, Los Angeles, Calif. Sponsored by the West Coast Comm., Los Angeles/Orange County Chapters, IEEE Computer Group, 26 May 1964.

Robbi, A.D. and Ricci, R. "Transfluxor Content-Addressable Memory", Proc. International Conference on Nonlinear Magnetics, April 1964.

Rogers, J. Algorithms for Complex Searches, presented at the IEEE Symp. on Search Memory, Los Angeles, California. Sponsored by the West Coast Comm., Los Angeles/Orange County Chapters, IEEE Computer Group.

Rogers, J.L. and Wolinsky, A. Associative Memory Algorithms and Their Cryogenic Implementation. Rept. 8670-6007-RU-000, prepared by Space Technology Labs., California.

Rosin, Robert F. "An Organization of an Associative Cryogenic Computer", Proc. SJCC. XXI, pp. 203-213, May 1962.

Rowland, C.A. and Berge, W.O. "A 300-Nanosecond Search Memory", Proc. Fall Joint Computer Conf., Las Vegas, Nev. (AFIPS No. 24): 59-65, November 1963.

Roy, Dr. Rob. "Recognizing Patterns with Threaded Cores", Electronic Industries, pp. 86-87, August 1964.

Rutman, R.A. An Algorithm for Placement of Interconnected Elements Based on Minimum Wire Length.

Salton, Gerard. "Associative Document Retrieval Techniques using Bibliographic Information", Journal of the Association for Computing Machinery. Volume 10, No. 4, pp. 440-457, October 1963.

Sams, Burnett, H. "On the Solution of an Information Retrieval Problem", Proc. SJCC. XXIII, pp. 289-97, Baltimore, Spartan Book, Inc. 1963.

Savitt, D. System Design of a Search Memory, presented at the IEEE Symp. on Search Memory, Los Angeles, Calif. Sponsored by the West Coast Comm., Los Angeles/Orange County Chapters, IEEE Computer Group, 26 May 1964.

Schick, Thomas, "Disk File Sorting", Comm. ACM. VI:6 pp. 330-31, June 1963.

Seeber, Robert R. "Symbol Manipulation with an Associative Memory", Proc. 16th National Meeting of ACM, Los Angeles, 5 September 1961.

Seeber, R. R. "Associative Self-Sorting Memory", Proc. Eastern Joint Computer Conf., New York City, N. Y. (NJCC No. 18):179-88, December 1960.

Seeber, R. R. Cryogenic Associative Memory, presented at Natl. Conf. ACM. Milwaukee, Wisc., August 1960.

Seeber, R. R., Jr. "Associative Self-Sorting Memory", Proc. E. J. C. C. XVIII, pp. 179-88, December, 1960.

Seeber, R. and Lindquist, A. Associative Logic for Highly Parallel System, Proc. Fall Joint Computer Conf., Las Vegas, Nev. (AFIPS No. 24):489-93, November 1963.

Seeber, R. R. and Lindquist, A. B. "Associative Memory with Ordered Retrieval", IBM J. Res. and Dev. VI: 1, pp. 126-36, January 1962.

Shahbender, R. et al. "Laminated Ferrite Memory", Proc. Fall Joint Computer Conference, Las Vegas, November 1963.

Shell, D. L. "A High-Speed Sorting Procedure", Communications of the ACM. II:7, pp. 30-32, July 1959.

Simmons, G. J. "Application of an Associatively Address Distributed Memory", Proc. Spring Joint Computer Conf., Washington, D. C. (AFIPS No. 25):493-511.

Sherry, Murray E. Memory Organization of a 7090 to Do Statistical Association Processing, presented at the American Documentation Institute, 27th ~~Annual Meeting~~, Philadelphia, Pa., 5-8 October 1964.

Simmons, G. J. A Mathematical Model for an Associative Memory, Sandia Corp., New Mexico, Rept. No. SCR-641, April 1963.

Singer, T. and Schupp, P. Associative Memory Computers from the Programming Point of View, Rept. No. ESD-TDR-63-245, prepared for Directorate of Computers, ESD, AFSC, Hanscom Field, by the Mitre Corporation, under Contract AF33(600)-39852, August 1963.

Singleton, Richard C. and Goldberg, J. "Random Selection Rates for Single-Field Superimposed Coding", Multiple Instantaneous Response File by J. Goldberg, et al. AD 266169. RADC-TR-61-233. Menlo Park, Stanford Research Inst., pp. 93-121, August 1961.

Slade, A. E. A Discussion of Associative Memories from a Device Point of View, presented at the American Documentation Institute, 27th Annual Meeting, Philadelphia, Pa., 5-8 October 1964.

Slade, A. E. "Proceedings of the International Symposium on the Theory of Switching, April 1957", Chapter in Harvard Computation Laboratory Series, 1959.

Slade, A.E. "A Cryotron Memory Cell", Proc. IRE. L:1, pp. 81-82, January 1962.

Slade, A.E. "The Woven Cryotron Memory", Proc. International Symp. on Theory of Switching, pp. 326-33, Cambridge, Mass., Harvard University Press, 1959.

Slade, A.E. and McMahon, H.O. "The Cryotron Catalog Memory System", Proc. E.J.C.C. Vol. 10, pp. 115-20, December 1956, Spec. Publ. T-92, (New York, American Inst. of Electr. Engineers, 1957).

Slade, A.E. and Smallman, C.R. Thin-Film Cryotron Catalog Memory, presented at Symp. Superconductive Techniques for Computing Systems, held May 1960. Published in Solid State.

Slade, A.E. and Smallman, C.R. Automatic Control. XIII:2, pp. 48-50, August 1960.

Slade, A.E. and Smallman, C.R. "Thin-Film Cryotron Catalog Memory", Solid-State Electronics. 1:4, pp. 357-362, 1960.

Slade, A.E. and Smallman, C.R. "Thin-Film Cryotron Catalog Memory", Proc. Symp. on Superconductive Techniques for Computing Systems, May 1960.

Slotnick, E.L.; and Borck, W.C.; and McReynolds, J.M. "The SOLOMON Computer", Proc. Fall Joint Computer Conf., Philadelphia, Pa. (AFIPS No. 22):97-107, December 1962.

Smallman, C.R.; Slade, A.E.; and Cohen, M.L. "Thin-Film Cryotrons", Proc. IRE. IIL:9, pp. 1562-82, September 1960.

Sohara, S. (Hughes Aircraft-Ground Sys. Divn.). Survey of Present and Potential Search Memory-Implementation and Techniques, presented at the IEEE Symp. on Search Memory, Los Angeles, California.

Stephens, P.A. AD 325 036. Culver City, Litton Systems, Inc., 51 pp., 12 May 1961. (SECRET report).

Stetsyura, G.G. "A New Principle for the Construction of a Memory Device", Dokl. Akad. Nauk SSSR, CXXXII (June 1960).

Tainiter, M. "Addressing for Random-Access Storage with Multiple Bucket Capacities", Journ. ACM. X:3, pp. 307-315, July 1963.

Thompson, Ramo Wooldridge Labs. True Content-Addressable Memory. Techn. Note 423-1. Canoga Park, TRW, April 1962.

TRW Space Technology Laboratories, Thompson Ramo Wooldridge, Inc., Redondo Beach, California. Computer Associative Memory Study - Final Report. Prepared for: Space Systems Division, Air Force Systems Command, Los Angeles, California, Contract No. AF04(695)-318, 15 July 1964.

Unger, S.A. "A Computer Oriented Toward Spatial Problems", Proc. IRE. Vol. 46, No. 10, pp. 1744-50, October 1959.

Slade, A.E. "A Cryotron Memory Cell", Proc. IRE. L:1, pp. 81-82, January 1962.

Slade, A.E. "The Woven Cryotron Memory", Proc. International Symp. on Theory of Switching, pp. 326-33, Cambridge, Mass., Harvard University Press, 1959.

Slade, A.E. and McMahon, H.O. "The Cryotron Catalog Memory System", Proc. E.J.C.C. Vol. 10, pp. 115-20, December 1956, Spec. Publ. T-92, (New York, American Inst. of Electr. Engineers, 1957).

Slade, A.E. and Smallman, C.R. Thin-Film Cryotron Catalog Memory, presented at Symp. Superconductive Techniques for Computing Systems, held May 1960. Published in Solid State.

Slade, A.E. and Smallman, C.R. Automatic Control. XIII:2, pp. 48-50, August 1960.

Slade, A.E. and Smallman, C.R. "Thin-Film Cryotron Catalog Memory", Solid-State Electronics. 1:4, pp. 357-362, 1960.

Slade, A.E. and Smallman, C.R. "Thin-Film Cryotron Catalog Memory", Proc. Symp. on Superconductive Techniques for Computing Systems, May 1960.

Slotnick, E.L.; and Borck, W.C.; and McReynolds, J.M. "The SOLOMON Computer", Proc. Fall Joint Computer Conf., Philadelphia, Pa. (AFIPS No. 22):97-107, December 1962.

Smallman, C.R.; Slade, A.E.; and Cohen, M.L. "Thin-Film Cryotrons", Proc. IRE. IIL:9, pp. 1562-82, September 1960.

Sohara, S. (Hughes Aircraft-Ground Sys. Divn.). Survey of Present and Potential Search Memory-Implementation and Techniques, presented at the IEEE Symp. on Search Memory, Los Angeles, California.

Stephens, P.A. AD 323 036. Culver City, Litton Systems, Inc., 51 pp., 12 May 1961. (SECRET report).

Stetsyura, G.G. "A New Principle for the Construction of a Memory Device", Dokl. Akad. Nauk SSSR, CXXXII (June 1960).

Tainiter, M. "Addressing for Random-Access Storage with Multiple Bucket Capacities", Journ. ACM. X:3, pp. 307-315, July 1963.

Thompson, Ramo Wooldridge Labs. True Content-Addressable Memory. Techn. Note 423-1. Canoga Park, TRW, April 1962.

TRW Space Technology Laboratories, Thompson Ramo Wooldridge, Inc., Redondo Beach, California. Computer Associative Memory Study - Final Report. Prepared for: Space Systems Division, Air Force Systems Command, Los Angeles, California, Contract No. AF04(695)-318, 15 July 1964.

Unger, S.A. "A Computer Oriented Toward Spatial Problems", Proc. IRE. Vol. 46, No. 10, pp. 1744-50, October 1959.

Tuttle, G. T. "How to Quiz a Whole Memory at Once", Electronics, Vol. 36, pp. 43-46, 1 November 1963.

Unger, S.H. "Pattern Detection and Recognition", Proc. IRE 47(10): 1737-52 October 1959.

Unger, S.H. "A Computer Oriented Toward Spatial Problems", Proc. IRE 46(10): 1744-50, October 1958.

Van de Riet, E.D. "Magnetic Realizations for MIRF Employing One Conductive Path per File Item", Multiple Instantaneous Response File by J. Goldberg, et al, pp. 158-193.

Wagner, E.G. and McCarthy, Tag Memory, U.S. Patent No. 3,093814, June 1963.

Wanlass, C.L. and Wanlass, S.D. "BIAX High-Speed Magnetic Computer Element", WESCON Convention Record Part 4, pp. 40-54, San Francisco, Calif., 18-21 August 1959.

Warheit, I.A. "The Direct Access Search System", AFIPS Conf. Proc. XXIV, pp. 167-172, November 1963.

Warren, D. High-Speed, Content Search in a Large, Rotating, Mass Memory, presented at the IEEE Symp. on Search Memory, Los Angeles, California. Sponsored by the West Coast Comm., Los Angeles/Orange County Chapters. IEEE Computer Group, 26 May 1964.

Weinstein, H. "Proposal for Ordered Sequential Detection of Simultaneous Multiple Responses", IEEE Trans. on Electronic Computers (Correspondence), Vol. EC-12, pp. 564-567, October 1963.

Wigington, R.L. "A Machine Organization for a General Purpose List Processor", IEEE Trans. Electronic Comp., EC-12(6): 707-14, December 1963.

Winkler, T. Semiannual Report on Digital Computer Systems Studies. STL/TR-60-0000-19224. AFBMD-TN-61-2. AD 257 621. Los Angeles, Space Technology Labs., 103 pp. December 1960.

Wise, Carl S. "Mathematical Analysis of Coding Systems", Punched Cards. Edited by R.S. Casey, J.W. Perry, et al. New York, Reinhold Publishing Co., 1958.

Wolinsky, A. Extreme Determination and Ordered Retrieval in Search Memories, presented at the IEEE Symp. on Search Memory, Los Angeles, Calif. Sponsored by the West Coast Comm., Los Angeles/Orange County Chapters, IEEE Computer Group, 26 May 1964.

Yang, C.C. and Lou, J.T. "Systematic Design of Cryogenic Logic Circuits," Proc. AFIPS 1964 Fall Joint Computer Conf. pp. 651-662.

Younker, E. L.; Heckler, C. H., Jr.; Masher, D. P.; and Yarborough, J. M. "Design of an Experimental Multiple Instantaneous Response File", Proc. AFIPS 1964 Spring Joint Computer Conference, Washington, D. C., April 1964. Review in Computing Reviews (ACM) Vol. 5 No. 6, pp. 400-401, November-December 1964, No. 6810.

Yovits, Marshall C. Large-Capacity Memory Techniques for Computing Systems. New York: The Macmillan Company, 440 pp. (ACM Monograph Series), 1962.

Zator Company. Papers and Publications: January 1957 - March 1960. Zator Tech. Bull. 129B. Cambridge, Zator Co., 3 pp., March 1960.

Unclassified
Security Classification

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Auerbach Corp Philadelphia, Pa.	2a. REPORT SECURITY CLASSIFICATION Uncl
	2b. GROUP

3. REPORT TITLE

Analysis of Small Associative Memories for Data Storage and Retrieval Systems

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Final Report, October 1964 to September 1965

5. AUTHOR(S) (Last name, first name, initial)

Green, Robert S., Mr.
Minker, Jack, Dr.
Shindle, Warren, E., Mr.

6. REPORT DATE July 1966	7a. TOTAL NO. OF PAGES 614 (124 + 490)	7b. NO. OF REFS 315
8a. CONTRACT OR GRANT NO. AF30(602)-3564	9a. ORIGINATOR'S REPORT NUMBER(S) 1231-TR2	
b. PROJECT NO. 4594		
c. Task 459402	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report) RADC-TR-65-397, (Vol I, Vol II)	
d.		

10. AVAILABILITY/LIMITATION NOTICES This document is subject to special export controls and each transmittal to foreign governments or foreign nationals may be made only with prior approval of RADC(EMLI),GAFB,N.Y. 13440.

11. SUPPLEMENTARY NOTES N/A	12. SPONSORING MILITARY ACTIVITY Rome Air Development Center (EMIID) Griffiss AFB,N.Y. 13440

13. ABSTRACT

The objective of this effort was to determine the effect of associative memories which are realizable today to aid in processing formatted record problems. The evaluation consisted of a comparison between the CDC 1604B and the CDC 1604B-Associative Memory to process the same problem. The Goodyear Aerospace Corp associative memory was used to establish state-of-the-art in associative memories, however, other associative memory designs were investigated.

DD FORM 1 JAN 64 1473

Unclassified
Security Classification

KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Associative Memories						
Content - Addressable Memories						
Comparison Evaluation						

INSTRUCTIONS

1. ORIGINATING ACTIVITY: Enter the name and address of the contractor, subcontractor, grantee, Department of Defense activity or other organization *(corporate author)* issuing the report.

2*a*. REPORT SECURITY CLASSIFICATION: Enter the overall security classification of the report. Indicate whether "Restricted Data" is included. Marking is to be in accordance with appropriate security regulations.

2*b*. GROUP: Automatic downgrading is specified in DoD Directive 5200.10 and Armed Forces Industrial Manual. Enter the group number. Also, when applicable, show that optional markings have been used for Group 3 and Group 4 as authorized.

3. REPORT TITLE: Enter the complete report title in all capital letters. Titles in all cases should be unclassified. If a meaningful title cannot be selected without classification, show title classification in all capitals in parenthesis immediately following the title.

4. DESCRIPTIVE NOTES: If appropriate, enter the type of report, e.g., interim, progress, summary, annual, or final. Give the inclusive dates when a specific reporting period is covered.

5. AUTHOR(S): Enter the name(s) of author(s) as shown on or in the report. Enter last name, first name, middle initial. If military, show rank and branch of service. The name of the principal author is an absolute minimum requirement.

6. REPORT DATE: Enter the date of the report as day, month, year; or month, year. If more than one date appears on the report, use date of publication.

7*a*. TOTAL NUMBER OF PAGES: The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

7*b*. NUMBER OF REFERENCES: Enter the total number of references cited in the report.

8*a*. CONTRACT OR GRANT NUMBER: If appropriate, enter the applicable number of the contract or grant under which the report was written.

8*b*, 8*c*, & 8*d*. PROJECT NUMBER: Enter the appropriate military department identification, such as project number, subproject number, system numbers, task number, etc.

9*a*. ORIGINATOR'S REPORT NUMBER(S): Enter the official report number by which the document will be identified and controlled by the originating activity. This number must be unique to this report.

9*b*. OTHER REPORT NUMBER(S): If the report has been assigned any other report numbers *(either by the originator or by the sponsor)*, also enter this number(s).

10. AVAILABILITY/LIMITATION NOTICES: Enter any limitations on further dissemination of the report, other than those imposed by security classification, using standard statements such as:

(1) "Qualified requesters may obtain copies of this report from DDC."

(2) "Foreign announcement and dissemination of this report by DDC is not authorized."

(3) "U. S. Government agencies may obtain copies of this report directly from DDC. Other qualified DDC users shall request through
______________________________."

(4) "U. S. military agencies may obtain copies of this report directly from DDC. Other qualified users shall request through
______________________________."

(5) "All distribution of this report is controlled. Qualified DDC users shall request through
______________________________."

If the report has been furnished to the Office of Technical Services, Department of Commerce, for sale to the public, indicate this fact and enter the price, if known.

11. SUPPLEMENTARY NOTES: Use for additional explanatory notes.

12. SPONSORING MILITARY ACTIVITY: Enter the name of the departmental project office or laboratory sponsoring *(paying for)* the research and development. Include address.

13. ABSTRACT: Enter an abstract giving a brief and factual summary of the document indicative of the report, even though it may also appear elsewhere in the body of the technical report. If additional space is required, a continuation sheet shall be attached.

It is highly desirable that the abstract of classified reports be unclassified. Each paragraph of the abstract shall end with an indication of the military security classification of the information in the paragraph, represented as *(TS)*, *(S)*, *(C)*, or *(U)*.

There is no limitation on the length of the abstract. However, the suggested length is from 150 to 225 words.

14. KEY WORDS: Key words are technically meaningful terms or short phrases that characterize a report and may be used as index entries for cataloging the report. Key words must be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context. The assignment of links, rules, and weights is optional