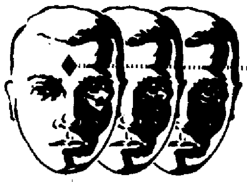
Proceedings of the WESTERN JOINT COMPUTER CONFERENCE



EXTENDING MAN'S INTELLECT

May 9-11, 1961 Los Angeles, California

Sponsors:

THE INSTITUTE OF RADIO ENGINEERS

Professional Group on Electronic Computers

THE AMERICAN INSTITUTE OF ELECTRICAL ENGINEERS

Committee on Computing Devices

THE ASSOCIATION FOR COMPUTING MACHINERY

Vol. 19 Price $4.00

PROCEEDINGS OF THE WESTERN JOINT COMPUTER CONFERENCE

PAPERS PRESENTED AT
THE JOINT IRE-AIEE-ACM COMPUTER CONFERENCE
LOS ANGELES, CALIF., MAY 9-11, 1961

Sponsors
THE INSTITUTE OF RADIO ENGINEERS
Professional Group on Electronic Computers
THE AMERICAN INSTITUTE OF ELECTRICAL ENGINEERS
Committee on Computing Devices
THE ASSOCIATION FOR COMPUTING MACHINERY

Published by
WESTERN JOINT COMPUTER CONFERENCE

ADDITIONAL COPIES

Additional copies may be purchased from the following sponsoring societies at $4.00 per copy. Checks should be made payable to any of the following societies:

INSTITUTE OF RADIO ENGINEERS
1 East 79th Street, New York 21, N.Y.

AMERICAN INSTITUTE OF ELECTRICAL ENGINEERS
33 West 39th Street, New York 18, N.Y.

ASSOCIATION FOR COMPUTING MACHINERY
2 East 63rd Street, New York 21, N.Y

Manufactured in the U.S.A. by
Griffin-Patterson Co., Inc., Glendale, Calif.

WESTERN JOINT COMPUTER CONFERENCE COMMITTEE

FOREWORD

The technical papers included in these Proceedings represent all the technical presentations made at the Western Joint Computer Conference. In addition, there is included a message from the Chairman of the National Joint Computer Committee, Dr. Morris Rubinoff. We are proud of the excellent contributions recorded here, so many of which directly support the 1961 WJCC theme "Extending Man's Intellect." Also, we are happy that we have been able to make these Proceedings available at the time of the Conference, thus enhancing the benefits of the Conference to registrants and making available the information in a timely manner.

It should be recognized, however, that the papers presented herein have not been selected by the usual procedures wherein a paper is refereed as to its appropriateness and edited for its content. Neither the NJCC nor the WJCC Committee can take responsibility for the accuracy of the facts or opinions expressed. We are confident, however, that the overwhelming majority of the papers presented here are responsible in all ways. Many papers were called but few were chosen; we are happy to record them here for the continuing advance and lasting annals of information processing technology.

WALTER F. BAUER
General Chairman
1961 Western Joint Computer Conference

MESSAGE FROM NJCC CHAIRMAN

This is an historic occasion. The close of this 1961 Western Joint Computer Conference will signal the change-over in administration of Joint Computer Conferences from the National Joint Computer Committee to the American Federation of Information Processing Societies (AFIPS), with broader scope and greater flexibility. As you know, AFIPS is a society of societies organized to represent through a single body the professional societies of the American computer and data processing world. The enthusiastic response to the formation of AFIPS is highly gratifying and lends encouragement, confidence and a sense of mission to those whom you have charged with conducting its activities.

There are times when the path to the future is best appreciated through a re-examination of the past. I would like to quote from a letter dated December 15, 1959, written by the late Chairman of NJCC, Professor Harry Goode, who contributed so much both to NJCC and to the birth of AFIPS:

"I believe the major objective in the formation of the society is to provide for *information flow* in all other instances than those provided for by the individual societies to their members.

"There are four types of such flow:

(1) Information flow between members of information processing societies nationally.

(2) Information flow between our national information processing society and foreign information processing societies.

(3) Information flow between societies in the information processing profession and other professions.

(4) Information flow from the information processing societies to the general and educational public.

"If we can recognize a firm set of objectives such as these (which of course need to be rewritten into a proper set of words), then what the society is to do is relatively clear-cut.

"The functions follow immediately from the objectives:

(1) Act as the American representative body on matters related to computing application and design, in a broad area of computational and information processing sciences.

(2) Advance the field by stimulating research into new aspects of computer sciences emphasizing the cross-pollination of ideas among member societies.

(3) Prepare, publish, and disseminate information of a tutorial nature to laymen, high school teachers and students, government offices and officials, etc.

(4) Maintain relations among American and foreign technical societies through conferences and symposia, cooperation with other societies in organizing sessions at their conferences, provide reference material to other societies on the computational sciences.

(5) Maintain membership in the International Federation of Information Processing Societies (IFIPS).

(6) Aid in certain actions of member societies involving participation and cooperation by more than one society.

(7) Sponsor the JCC's."

The Constitution of AFIPS reflects these views in their entirety. With your frequently demonstrated cooperation and support, the Board of Governors of AFIPS will continue to conduct our successful Joint Computer Conferences and to represent the United States in our International Federation, IFIPS. As new societies join the Federation, it will gradually provide the hoped-for broad representation of the American information processing profession. We will seek to establish AFIPS as the information center on data processing including not only bibliographies of written material, but also a calendar of events of computer activities in the United States and throughout the world, a roster of individuals active in information processing, and a current file of developments in progress or recently consummated. We plan to establish a speakers' bureau to carry information on the information processing field to educational institutions and professional societies. We plan to establish a public information committee which, through the media of personal contacts, press releases and tutorial articles, will make available to laymen, to government agencies, to affiliated and member societies and to the profession as a whole, the present status and the probable future of information processing in the United States.

I trust that with your continued cooperation and support our efforts will meet with a long string of successes.

Respectfully submitted,

Morris Rubinoff, Chairman
National Joint Computer Committee

TABLE OF CONTENTS

TABLE OF CONTENTS, continued

TABLE OF CONTENTS, continued

SIMULATION: A SURVEY

Harry H. Harman
System Development Corporation
Santa Monica, California

Introduction

Simulation may be traced back to the beginning of time -- be it the make-believe world of the child at play,or the adult make-believe world of the stage. The impetus for modern scientific simulation came with the development of analog computers in the 1930's; and progressed even further when the electronic digital computers were created. The very definition of an analog computer contains the notion of simulation, viz., a device which simulates some mathematical process and in which the results of this process can be observed as physical quantities,such as voltages, currents, or shaft positions. While there is no doubt that the analog computer represents one aspect of simulation, the truly new simulation advances came with the digital computer. In the past two decades, since the development of Mark I by Howard Aiken and since Eckert and Mauchly designed the ENIAC, tremendous strides have been made in science and technology ascribable directly to the flourishing new computing discipline.

The revolutionary impact of the electronic computer on our society may well be equal to that of atomic energy -- and may actually surpass it in the long run. A direct consequence of the computer is the burgeoning activity which collectively goes under the name, "simulation". The growing awareness, and popularity of this field of activity is evidenced by a recent article in Business Week in which a parallel is drawn between the group of simulation experts and the group of painters known as the Futurists. Just as the art works might bear no direct resemblance to the subjects for which they were named, so the mathematical formulas, flow diagrams, and computer outputs bear no direct resemblance to the physical world which they simulate. Moreover, this symbolic art "represents a massive assault on tradition -- in this case, the traditional art of managing large organizations."[16] This assault -- involving scientific systems analysis and simulation techniques -- first occurred on military systems problems, but more recently has found its way into business and industrial systems problems as well.

Definitions

To appraise the current work in simulation and to apprise you of the general status of this subject is the raison d'être of this session. In my review of the work in this area, I came across John Harling's paper, "Simulation Techniques in Operations Research -- A Review".[5] From the title it would appear that my work had been done for me. His opening remarks draw attention to the fact that "simulation" is a somewhat ill-defined subject and that considerable confusion exists in the terminology employed, and he goes on to say: "The term 'Monte Carlo' is presently somewhat fashionable; the term 'simulation' is to be preferred, because it does not suggest that the technique is limited to what is familiar to statisticians as a sampling experiment." (p.307) He equates "simulation" with "Monte Carlo methods" and thereby implies a much more restrictive usage of simulation than is intended in the present Survey.

The term "simulation" has recently become very popular, and probably somewhat overworked. There are many and sundry definitions of simulation, and a review and study of some of these should help us gain a better perspective of the broad spectrum of simulation. Webster only provides the fundamental notion that simulation is an act of "assuming the appearance of, without the reality". Thomas and Deemer[20] suggest the following paraphrase of Webster: "to simulate is to attain the essence of, without the reality." Note that the substitution of "essence" for "appearance" makes the vital difference between the scientific and the casual use of simulation. It not only is not necessary that the simulator not "appear" as its real-life counterpart, but frequently attempts to imitate reality closely may be detrimental to the purposes of the simulation. For example, to expedite the training of pilots a relatively accurate duplication of the cockpit is necessary for the trainer, but to duplicate the bulky whole of the airplane would defeat the purpose of the simulator. Thomas and Deemer advise that "we should deplore the tendency to introduce trappings and ornaments in simulation to gain the 'appearance' of reality when it is the 'essence' which we need." (p.5)

In a technical dictionary[7] the term "simulator" is defined as follows:

> A physical system which is analogous to a model under study (as, for instance, an electric network in which the elements

> are in correspondence with those of an economic model). The variables of interest in the model appear as physical variables (such as voltages and currents) and may be studied by an examination of the physical variables in the simulator. (p.267)

This definition covers what we normally would consider simulation when accomplished by analog or digital computers. Nonetheless, it is not the universally accepted definition, alternatives being proposed by practically each separate field of application.

Thus, in the area of Operations Research, Harling[5] states: "By simulation is meant the technique of setting up a stochastic model of a real situation and then performing sampling experiments upon the model. The feature which distinguishes simulation from a mere sampling experiment in the classical sense is that of the stochastic model". (p.307) As noted above, this definition of simulation is equivalent to the Monte Carlo technique; and is, in fact, almost identical with the definition of the latter provided by A.S. Householder:[6]

> The Monte Carlo method may briefly be described as the device of studying an artificial stochastic model of a physical or mathematical process.... The novelty [of the Monte Carlo method] lies rather in the suggestion that where an equation arising in a nonprobabilistic context demands a numerical solution not easily obtainable by standard numerical methods, there may exist a stochastic process with distributions or parameters which satisfy the equation, and it may actually be more efficient to construct such a process and compute the statistics than to attempt to use those standard methods (p.v).

While this represents a very powerful and useful technique in simulation, Monte Carlo does not encompass all the legitimate scientific aspects of simulation.

In their book, <u>System Engineering</u>,[4] Goode and Machol give a half dozen or more examples of simulation in which the Monte Carlo Method is used in queueing problems. They do not, however, take the foregoing definition. Instead, they define simulation to be "the study of a system by the cut-and-try examination of its mathematical representation by means of a large-scale computer". (p.403) While some people (not in this audience) might object to the qualifier that a "large-scale computer" be the means of the study, they would certainly grant its <u>modus operandi</u>. This is an operational definition, and as such it proposes more or less exact procedures to be followed in executing a program of simulation. Specifically, Goode and Machol propose a series of steps (pp.404-7) including the choice of computer (analog or digital, in particular); construction of the computational flow diagram (it being assumed that the mathematical model of the system has been formulated); determination of preliminary (analytical) solutions; choice of cases to be treated, with a view toward reducing the number of runs; data reduction and analysis (some to be done run by run); and consideration of the simulation of human beings (by some simple analytical function or by actual inclusion in the simulation).

A type of working definition is proposed in the field of Management Science. Here, simulation is conceived as "the science of employing computational models as description for the purposes of (1) learning, (2) experimenting, (3) predicting in management problems."[19] A similar definition, which more specifically delimits the area of consideration, is the following:[1]

> The systematic abstraction and partial duplication of a phenomenon for the purposes of effecting 1) the transfer of training from a synthetic environment to a real environment; 2) the analysis of a specific phenomenon; or 3) the design of a specific system in terms of certain conditions, behavior, and mechanisms.(p.6)

The behavioral scientist, accumstomed to laboratory experimentation, puts it even more directly:[8] "By simulation, we mean a technique of substituting a synthetic environment for a real one -- so that it is possible to work under laboratory conditions of control."

The foregoing definitions range in emphasis from a sampling plan (which distorts distributions in order to obtain relatively efficient estimates of the parameters) and the mere use of a large-scale computer, to a simple delineation of the area of inquiry. What they have in common is an attempt to substitute other elements for some or all of the real elements of a system. Perhaps the simplest and most direct definition of simulation is merely the <u>act of representing some aspects of the real world by numbers or other symbols that can be easily manipulated in order to facilitate its study.</u> In this sense, simulation is one of the oldest analytical tools.

<u>Classifications</u>

However simulation is defined, there remains the problem of selecting the appropriate elements of a system to be simulated. Which aspects are represented, and how they are represented, constitute the distinguishing characteristics of the different types of simulation. Hopefully, these considerations should also provide for the meaningful classification of simulation types.

After an exhaustive search of the literature, and several months' cogitation, the writer was reluctantly forced to conclude that there is no completely adequate taxonomy of simulation types. Perhaps some day a reasonable basis will evolve for classifying simulation types into major and subordinate categories, and the practitioner will be assisted thereby; but at the present time, we can do very little in that direction.

About the best that has been proposed (see for example, I. J. Good[3]) is a single continuum on which the model is classified according to its degree of abstraction from the real-life system, operation, or procedure. Thus, the focus is on the simulation model and its relationship to its real-life counterpart. This conceptual basis for ordering simulation types follows:

(1) In the most extreme instance (ultimate or trivial, depending on your point of view), the real system can be used as the "model" to gain knowledge about itself. However direct and simple it might sound, it is usually neither practical nor feasible to determine the inherent properties of a system by observing its operations. Limited time and resources often force the use of shorter, less expensive methods than the "identity simulation".

(2) Only one step removed from the real-life instance is the attempt to replicate it with the highest degree of fidelity, by means of an operational model of the system in its normal environment. A SAC mission flown to test the air defenses of the United States is an example of an essential replication of a war situation. Enemy bombers are replaced by SAC bombers; ADC fires no weapons. Such "replication simulation" really involves very little abstraction from reality,and also provides very little gain; except to make possible the limited study of selected dangerous or future situations. A subcategory of this classification might involve essential replication of operational gear while employing abstracted inputs.A case in point is the Air Defense Command's System Training Program (discussed below).

(3) Next, along our continuum, the replication might be attempted in the laboratory instead of in the field. Here it is necessary to choose the relevant features of the real system for representation in the laboratory, and also to decide on the means of such representation. A system may be made up of such diverse elements as people, hardware,operating procedures, mathematical functions, and probability distributions. A laboratory model might consist of the actual replication of some elements and the abstraction and substitution by symbolic representation of others. It should be noted that every kind of substitution is possible: people are often simulated by hardware, but the reverse is also done. A wide range of simulation types is encompassed by "laboratory simulation", and perhaps is best exemplified by operational gaming.

(4) More clear-cut abstraction from reality is involved in the complete "computer simulation" of a real system. In some circles this is the only admissable type of simulation. There is no room for human beings or real hardware components in this model of the system. All aspects of the system must be reduced to logical decision rules and operations which can be programmed. If the model of the system consists only of mathematical functions, the simulation is said to be <u>deterministic</u>. If it also includes probability <u>distributions</u> then it is <u>stochastic</u>. This type of simulation is quite common in operations research, with a popular example being a "computer simulation" of a (hypothetical) business firm.

(5) The highest degree of abstraction leads to the complete "analytical simulation", wherein the real system is represented completely by means of a mathematical model and a solution (at least theoretically) can be obtained by analytical means. Essentially, the problem here is that of solving a set of equations. Even if a closed form is not available, approximate methods (including Monte Carlo) can be employed to get a solution. The least and the highest degrees of abstraction -- "identity simulation"and complete "analytical simulation"-- may not be of much experimental value, but they do provide useful conceptual bounds for the simulation continuum.

<u>Need for further classification</u>.-- While the foregoing considerations provide a fundamental (philosophical) continuum on which simulation types might be ordered, it is not sufficiently discriminating. The bulk of the simulation studies reported in the literature would fall into one or two categories only. Further, more detailed distinctions could lead to generalized principles and thus to the full development of a discipline of simulation. The additional dimensions of simulation cannot be adequately determined at the present rudimentary stage of development of this field.

<u>Dichotomous classifications</u>.-- What is frequently done as an alternative is to break the total field of simulation into two classes. Commonly encountered examples of such dichotomy, or polarity, is deterministic-stochastic; deductive-inductive;analytical-physical;computerized-manual; or one of the many variants of these. An important consideration is the absence or presence of at least one human being in the simulated model. While this seems to offer a real distinguishing characteristic, it does not help nearly as much as anticipated. There can still be stochastic models which are simulated entirely in a computer, or by means of a computer and people. For this reason, the writer discarded an earlier plan in which the primary dichotomy was into "automaton-simulation" and "bio-simulation". Differences in simulations that are fully computerized and those that involve human beings may be useful,but should probably be subordinated to more fundamental classification concepts.

Even this crude classification scheme may provide a useful guide in planning a simulation experiment. As a general rule, increasing experimental control can be attained by moving in the direction of a complete mathematical model, but unfortunately this usually is associated with decreasing realism. The more that is known about the properties of an element of a system,the better it can be simulated. Imperfectly understood system elements probably should be used " as is" in the model rather than approximated in a probabilistic manner or by decision rules. Adequate simulation of a system in the laboratory requires a detailed systems analysis with particular attention paid to the functional structure of the various tasks and the operations to be performed by the human beings in the system. Since the human actions are certainly of a stochastic nature,realistic simulation of a man-machine system can best be accomplished by having the human elements in the model.

Classification by objective.-- An alternative breakdown of simulation activities can be made according to the purpose or objective of the simulation. The principal categories usually employed are evaluation, training, and demonstration. With the emergence of very large military command and control systems, the old trial-and-error method had to give way to simulation as the primary technique for the design and development of such systems, as well as for the evaluation of alternative solutions to system problems. Again, in the implementation and operation of such systems, simulation has been found to be a very effective device for training. Not only have simulators been employed for individual flight instruction in place of expensive and dangerous procedures, but similar efficiencies have been realized in training groups in total system operations through simulation. This is one of the chief objectives of management games as well as the specific training programs of military systems. In the demonstration role, simulation serves as a means of indoctrination -- to exhibit the feasibility of a complex system.

Simulation as a research tool

While this very brief account of the uses of simulation for evaluation,demonstration,and training immediately points up its value,some more definite indication of the advantages of simulation as a research tool in the study of complex systems seems to be in order. First of all, the real system in the field is not as amenable to control as a simulation of it. At the same time there is no interruption of the on-going activities in order to conduct the research.Also, productive research requires the taking of quantitative measurements, which again can better be accomplished in a simulation study than by observation of the actual system.

These primary advantages are really the advantages of the laboratory over the field,regardless of whether it is a chemistry laboratory or a digital-computer laboratory. Simulation as a research technique has more specific advantages:

(1) It can compress or expand real time. A business operation of a year can be simulated in minutes in order to study long term trends or to study the operations under varying alternatives. On the other hand, the process can be slowed down to permit the more detailed study of critical situations.

(2) It provides the ability to experiment, test,and evaluate new systems or proposed changes to existing systems in advance of having to make firm commitments.Aside from great economy of time, simulation of this type makes it possible to consider hypothetical systems which may be dangerous or impossible to try any other way.An interesting example involves the procedure the Cornell Aeronautical Laboratory employed in designing and constructing the Mark I perceptron for the automatic identification of simple patterns. They first demonstrated by simulation on a computer (IBM 704) that such an experimental machine could be built.

(3) It makes for more economical experimentation, both in time and money. A complete "computer simulation" of a system usually can be run in very short time once the program has been developed. However, the cost of creating a large-scale computer simulation program can be prohibitive. Usually it is justified because of continued experimentation with the model, but on occasion the payoff may be so great as to justify even a single trial.

(4)It permits the replication of experiments under different conditions. An important example is the replication of economic time-series, which just could not be accomplished without simulation.

Review of simulation activities

Extent of literature.-- The acceptance of simulation evidently has been widespread -- as witness the increasing number of simulation studies in the last decade. Prior to 1951 there was nothing in the scientific literature on this subject. The most recently published bibliography[15] contains 344 entries (including 6 other bibliographies)and except for one reference ("A Simplified War Game, 1897) the earliest article is dated 1951. Two other bibliographies merit special mention. Malcolm[11] presents what he terms "a fair sampling of simulation literature to date". Concerned primarily with the application of simulation to management problems, he subdivides the 165 titles into industrial and military applications and separates simulation games from the rest. The other,[12] while not specifically addressed to simulation,presents 477 references to the closely allied subject of systems research. One of the interesting aspects of the latter bibliography is that it also contains a topical outline of the field and each reference is assigned to one or more of the classification categories. The extent of the literature on simulation has grown to such immense proportions, in so short a time, that the truly scholarly exploration of this field looms as a formidable effort for all but the most serious student.

No attempt will be made here to review the content of different simulation studies. The objective is only to indicate the scope of such studies. One such collection of 17 studies appears in the "Report of Systems Simulation Symposium", published in 1958. These include typical inventory-control, scheduling, cargo handling, and waiting-line problems on the industrial side; related problems on logistics systems peculiar to the military, as well as military "laboratory simulations", incorporating systems of men and equipment; and even some methodological considerations directed at increasing the speed of simulation and statistical problems associated with Monte Carlo sampling.

As regards the technical aspects of simulation, the results of current research activities appear, principally, in the Operations Research journal, specialized statistical journals, and publications of various research institutes. Of special interest is the report of the first Symposium on the Monte Carlo Method[6] and two subsequent symposia [17,18] on the same subject.

Operational gaming.-- The simulation studies that have attracted the most attention in recent years may be described by the generic term "games" -- intended to cover such activities as war gaming, business management games, and operational gaming in general. In their excellent article, Thomas and Deemer[20] first distinguish the basic concepts of simulation, Monte Carlo, and operational gaming; present a brief review of some of the theory of games of strategy; and then compare the approaches of gaming and non-gaming techniques to competitive situations. The role of operational gaming is best expressed in their words:

> Although simulation and Monte Carlo methods are often used in gaming we feel that the essence of operational gaming lies rather in its emphasis on the playing of a game. There is playing to formulate a game, playing to solve a game, and playing to impart present knowledge of a game. Thus we define operational gaming as the serious use of playing as a primary device to formulate a game, to solve a game, or to impart something of the solution of a game. (p.6)

In practical applications, the technique of gaming is aimed principally at providing practice in working through alternative sequences in considerable detail. Within the framework of a particular game certain input parameters can be altered to provide innumerable variations. When human teams participate in such games, they not only gain practice in comprehending the consequences of particular moves and sequences of events, but also gain some insight into the perspective of the participants.

The development and present usage of management games is reviewed by Joel Kibbee in the following paper on this Program. He stresses the importance of computers in this area, and discusses the building of models and programming of management games. It should be remembered that non-computer or manual business games (e.g., as developed by Stanley Vance at the University of Oregon and by John L. Kennedy at Princeton University) have considerable merit as tools for management training and development as well.

Management control.-- Perhaps one of the most powerful tools for management control of large-scale programs is the activity known as PERT (Program Evaluation Review Technique). This system of charting the key milestones into a network for the accomplishment of an objective, dependent on many and diverse factors, was first developed in conjunction with the Polaris program.[10] As a result of such management control, the Polaris program became operational two years earlier than originally anticipated. A similar technique developed for the Air Force by Douglas Aircraft Company in conjunction with the Skybolt program is PEP (Program Evaluation Procedure). The PERT/PEP program evaluation techniques now are being extended to almost all Army, Navy, and Air Force weapons systems.[9] Among other computer-based methods for monitoring schedules being developed is SCANS (Scheduling and Control by Automated Network Systems) at System Development Corporation. The aspect of these techniques which is especially germane to this Session is the optimization of networks through simulation. By devising a "computer simulation" of the scheduling technique, alternative management decisions can be tried, and from the output an optimal solution can be determined. Closely related to these types of programs is the Decision Gaming work on which Dr. Vazsonyi reports later in this Session.

Social behavior.-- Turning to another area, Ellis Scott[14], calls attention to dozens of studies on simulation of social processes being carried out in universities and research laboratories from coast to coast. His survey is concerned with research in the behavioral sciences which use computers in the simulation of social behavior. The studies range from experiments in interactions and conformity of small groups to intergroup relations in the community to the behavior of an entire society and international relations.

Vehicular traffic.--Still another area which is receiving more and more attention is that of vehicular traffic control. While the earliest works, by H. H. Goode, G. F. Newell, and others, only date back about six years, the activity has been gaining considerable momentum since then.

Research is going on in all parts of the country. The extent of the national interest is evidenced by the conference on transportation research convened by the National Academy of Sciences last fall. About 150 participants from government, industry, universities, and research institutions met to review and formulate a program of research on transportation in the United States. A more recent conference[13] was devoted exclusively to the utilization of simulation as a research tool in the areas of highway and vehicle improvement, traffic control and enforcement, and driver and safety education.

An example of a physical model for studying driver performance, car construction, and road design is the "driving simulator" at the UCLA Institute of Transportation and Traffic Engineering. The cab of this simulator consists of a standard station wagon on a treadmill of steel rollers, which faces a 10-ft high semicircular screen and with a small screen on the car's rear window. Movie projectors throw traffic scenes on both screens and a battery of instruments record changes in steeringwheel movement, acceleration, braking, and in the driver's breathing rate and in emotional stress.

Although the ultimate goal is to consider the total system, including the driver and the traffic, at this stage of development of methodology, it seems wise to distinguish "driving simulation" from "traffic simulation". Early work on traffic simulation was restricted to one or two lanes of very short stretches of highway, and required inordinate amounts of computer time. Nonetheless, such work pointed to the feasibility of running simulation studies of traffic flow. A much more extensive model of expressway traffic flow has been developed at the Midwest Research Institute, and is reported by Glickstein and Levy later in this Session.

Simulation in man-machine laboratory research

The foregoing review points to many exciting and challenging activities -- emerging as a result of the development of the digital electronic computer, the use of simulation, and the increased awareness of the "systems approach". Thus, the study of large, complex man-machine systems has become possible.

Just as trial-and-error experimentation has been a respected technique in the development of the classical sciences, so in the study of complex systems the new techniques of simulation may be employed to explore and to define the problem itself. The direction and course of study of a man-machine system should be permitted (at least in the early stages) to be altered and restructured during the simulation and according to insights gained from the simulation itself. This use of simulation as a new kind of research tool is perhaps the outstanding feature of such laboratories as RAND's Logistics Systems Laboratory and SDC's Systems Simulation Research Laboratory discussed below.

NEWS.-- Entire laboratories have been built to exploit simulation for teaching purposes and evaluation of systems. Perhaps the first such facility to be conceived (in 1945), but which was not funded until 1950 and then took eight years to build, is the simulator at the U.S. Naval War College at Newport, Rhode Island. This facility and the exercise conducted in it is called NEWS (Naval Electronic Warfare Simulator). At the heart of the system is a very large analog computer (known as the Damage Computer) which is designed primarily to assess damage and to provide feedback to the several forces playing, to indicate their remaining effectiveness. The exercise is primarily a training device -- used in war gaming, in the final stages of tactical training of naval officers from the fleet.

SRL.-- Another laboratory in which simulation was employed as the principal tool was the Systems Research Laboratory (SRL) of The RAND Corporation. From 1951 to 1954 this laboratory employed simulation to generate stimuli for the study of information processing centers. The essential features of a radar site were created in the laboratory and by carefully controlling the synthetic inputs to the system and recording the behavior of the group it was possible to study the effectiveness of various man-machine combinations and procedures.

STP.-- The research in SRL eventually gave rise to the Air Defense Command's System Training Program (STP) -- probably the largest-scale simulation effort ever attempted. STP is now in operation throughout the United States, as well as in Alaska, Canada, and Europe. Training exercises are conducted in the normal working environment at the radar sites, direction centers in the SAGE system, Division Headquarters, and higher commands. Fundamental to this vast program is the creation of problem materials by means of an IBM 709 and special off-line and EAM equipment. Through these means synchronized radar pictures for large areas of the country are simulated along with other inputs required by the operating system, e.g., flight plan information, intelligence and weather information, and commands from higher headquarters. Also, various lists and maps are prepared for the trainers to assist them in observing and recording crew actions in order to furnish feedback on system performance to the crew immediately after each exercise. Through simulation of this type it is possible to provide exercise of air defense procedures and regulations, applicable either in peace or in war situations, at a fraction of what it would cost with "replication simulation".

LSL.-- In 1956, the Logistics Systems Laboratory (LSL) was established at RAND under Air

Force sponsorship. The first study in this laboratory involved the simulation of two large logistics systems for purposes of comparing their effectiveness under different governing policies and resources. The system consisted of men and machine resources together with policy rules on the use of such resources in simulated stress situations such as war. The simulated environment required a certain amount of aircraft in flying and alert states while the systems' capability to meet these objectives were limited by malfunctioning parts, procurement and transportation delays, etc. The human participants represented management personnel while higher echelon policies in the utilization of resources were simulated in the computer. The ultimate criteria of the effectiveness of the systems were the number of aircraft in commission and dollar costs. While the purpose of the first study in LSL was to test the feasibility of introducing new procedures into an existing Air Force logistics system and to compare the modified system with the original one, the second laboratory problem has quite a different objective. Its purpose is to improve the design of the operational control system through the use of simulation. The complete description of this study is presented by Dr. Steger later in this program.

ASDEC.-- A somewhat different type of facility in which simulation is employed to test and evaluate electronic systems is the Applied Systems Development Evaluation Center (ASDEC) of the Naval Electronics Laboratory at San Diego. Recently the Navy Tactical Data System was being evaluated. The operational system was simulated by means of actual hardware components such as the Univac M460 computer and cardboard mockups of display and control equipment. The facility includes an analog-to-digital computer which generates synthetic radar data used in the testing of operational systems.

NBS Study.-- Perhaps the largest single step in the exploitation of simulation for research purposes was the recent Feasibility Study[2] conducted by the National Bureau of Standards. The broad objectives of this study are best indicated in its opening paragraph:

> This report presents the results of a study of the feasibility, design, and cost of a large-scale tool to be used in a research program on man-machine systems. This tool facilitates the simulation of complex weapon systems for purposes of laboratory experimentation with human subjects in the system feedback loops. It is intended to aid in the optimization of system performance through studies of man-machine dynamics. It incorporates capabilities which represent a substantial advance over those of existing facilities for research on man-machine systems.

Feasibility was demonstrated through the actual design, implementation and operation of a scale model of the desired facility. The work done at the National Bureau of Standards provided the fundamental guidelines and philosophy for the more ambitious laboratory facility being built by the System Development Corporation in Santa Monica.

SSRL.-- Recognizing the importance of recent work in simulation, as well as recognizing the need for continued and expanded support for the further development of this area, with particular emphasis on its use in the study of complex man-machine systems, SDC decided to create a general-purpose, computer-based, facility in which such research could be conducted. Plans for the Systems Simulation Research Laboratory (SSRL) were initiated about fifteen months ago and are about to come to fruition. My report on SSRL is in greater detail because of my involvement and familiarity with it.

The physical facility, covering about 20,000 square feet, has just been completed. The main experimental operations space is a room approximately 45 x 50 feet with 20-foot clearance from floor to ceiling. It is completely surrounded by an elevated observation area. This large room may be divided into appropriate smaller areas by means of movable walls. Adjacent to the large, high-ceiling space are smaller, standard height experimental areas, which also may be adjusted in size and shape to accommodate the operations and observation requirements of specific projects.

A basic concept in planning a laboratory of this kind is the distinction between universal-type and project-specific type equipment. Of the former type, the most important is the general-purpose digital computer. A Philco 2000 system was selected and is now installed. Another major piece of equipment is a transducer that permits human beings and other real-time elements of a system to communicate with the computer. Such a real-time switch and storage unit (RL-101) has been designed and built at SDC and will be ready for integration with the computer next month. An internal telephone system (up to 120 stations), a public address system, recording facilities for any audio line, and a closed-circuit television system round out the general-purpose equipment of the laboratory at this time. The specific hardware requirements for the first couple of projects are now being determined.

Another basic concept is a general-purpose programming system. Perhaps some day we will have a general-purpose simulation program which will greatly facilitate the execution of research projects. For the present, however, we refer to the basic utility program system for the Philco 2000 operating with the RL-101. At SDC we are using a problem oriented language, known as JOVIAL, which is patterned after Algol (the International Algebraic Language). The principal effort involves the preparation of a JOVIAL Translator for the Philco 2000; but which has been written in such a manner that preliminary testing and actual compilation could be done on an IBM 709. Also, an executive control program has been developed which takes cognizance of the requirements introduced by the RL-101 and of the unusual nature of the applications of the Philco 2000. The programming for the initial research projects is proceeding concurrently with the utility programming.

The new laboratory is expected to enhance the present research efforts of SDC and to open entirely new avenues of research endeavor. In the former category are a number of research projects that have necessarily been limited in scope, but which can now be broadened because of the new facilities. One such area is that of automated teaching. Successful research in this area has been conducted at SDC in the last two years, but the constraint of a single student to the teaching machine has been a severe limitation. This made the gathering of statistical data very time-consuming. Also, any potential application of automated teaching techniques in the academic or the military or industrial organizations would certainly require more efficient means than individual tutoring. Thus, the next stage in this research effort is to create a Computerized Laboratory School System (CLASS) which project will be studied in SSRL very soon.

Another example of present research at SDC which can be expanded through the medium of the new laboratory is the study of Management Control Systems. At the present time, the research consists of a "computer simulation" of the behavior of a business system. This model enables the study of the reaction of the organization to specific changes under alternative sets of decision rules. As interesting as the computer simulation might be, it will be found lacking in a basic ingredient insofar as acceptance by real-world managers is concerned. That ingredient is the true human variability in decision making.The particular model certainly can be made more valid -- albeit, more complex and less controllable -- by introducing human decision makers at certain critical points in place of decision rules. Such a "laboratory simulation" model, at a later phase of the research, will be possible in the SSRL.

The first new research endeavor to exploit the SSRL facilities is a study of a terminal air traffic control system operating in a post-1970 air environment. Projected increases in traffic volume and aircraft speeds indicate that terminal control zones will increase in size and will therefore include many airports within a single complex. Coordination among many airports of the control of high density traffic of widely differing performance characteristics poses significant problems of organization and planning. It is believed that in order to effect the safe, orderly and expeditious flow of air traffic in a terminal complex,there will be a need for a new planning agency in addition to the control agencies in intimate contact with the details of the environment. The general purpose of this project is to investigate the functional interactions among the control agencies, and to evolve alternative hypotheses regarding superordinate planning agencies.

In the first phase of the project the configuration simulated is an air traffic control system for a two-airport terminal complex. The system consists of the operators and equipment representing the following agencies for each of the two airports:Stack Control,Approach Gate Control, Approach Control, Departure Control, and Flight Data Processing. Some of these agencies include human operators while others are represented by completely automatic processes. The objectives of Phase I are to study inter-airport coordination problems and to identify significant variables for future systematic investigation. Additional planning and coordination functions will be added in subsequent configurations as they are indicated by Phase I results. This project -- involving a "laboratory simulation" model --is an excellent example of the utilization of the best aspects of the broad range of simulation techniques in order to experiment with a complex man-machine system.

* * *

In this wide range of simulation work which we have reviewed two distinct activities stand out, neither one taking much cognizance of the other. On the one hand, simulation work is being done in the Operations Research field which may be classified largely as "computer simulation". On the other hand, there is the group of behavioral scientists, experimental psychologists in particular, engaged in the simulation of environmental conditions which may be called "laboratory simulation". Each of these groups could learn a great deal from the other. Furthermore, there is increasing evidence that "pure" simulation will have to be modified if it is to stand the test of validation. What is necessary is the marriage of the two approaches -- a realistic possibility in the new man-machine system laboratory.

References

1. Bogdanoff, E., et al. Simulation: An Introduction to a new Technology. TM-499. Santa Monica,Calif.: System Development Corp., 1960.

2. Ernst, Arthur A. Feasibility Study for a Man-Machine Systems Research Facility. WADC Technical Report 59-51. ASTIA Doc. No. AD 213589.

3. Good, I. J. Discussion in "Symposium on Monte Carlo Methods," J.Royal Stat.Soc.,B 16(1954), 68-69.

4. Goode, Harry H., and Machol, Robert E. System Engineering. New York, N.Y.: McGraw-Hill Book Co., Inc., 1957. Pp. xii + 551.

5. Harling, John."Simulation Techniques in Operations Research - A Review," Operations Research, 6 (1958), 307-19.

6. Householder, A.H.,Forsythe,G.E., and Germond, H.H. (eds.). Monte Carlo Method. Natl. Bur. of Standards Applied Math.Series,No.12, 1951.

7. Kendall, Maurice G., and Buckland, William R. A Dictionary of Statistical Terms. New York, N.Y.: Hafner Publishing Co.,1957.Pp.ix + 493.

8. Kennedy, J.L., Durking, J.E., and Kling, F.R. "Growing Synthetic Organisms in Synthetic Environments." Unpublished Paper, Princeton University, 1960.

9. Klass, Philip J. "PERT/PEP Management Tool Use Grows," Aviation Week, Vol.73, No. 22 (Nov. 28, 1960), 85-91.

10. Malcolm, D.G., Rosebloom, J.H., Clark, C.E., and Fazar, W. "Application of a Technique for Research and Development Program Evaluation," Operations Research, 7 (1959), 646-69.

11. Malcolm, D. G. "Bibliography on the Use of Simulation in Management Analysis,"Operations Research, 8 (1960), 169-77.

12. McGrath, Joseph E., and Nordlie, Peter G. Synthesis and Comparison of System Research Methods, App.B (1960). ASTIA No. AD 234463.

13. National Conference on Driving Simulation. Sponsored by Automotive Safety Foundation, Public Health Service. Santa Monica, Calif., Feb. 27 to March 1, 1961.

14. Scott, Ellis. Simulation of Social Processes: A Preliminary Report of a Survey of Current Research in the Behavioral Sciences. TM-435. Santa Monica,Calif.:System Development Corp., 1959. Pp. 15.

15. Shubik,Martin. "Bibliography on Simulation, Gaming, Artificial Intelligence and Allied Topics," J. Am. Stat.Assoc.,55 (1960),736-51.

16. Silk, L. J. "The Gentle Art of Simulation," Business Week, Nov. 29, 1958, 73-82.

17. "Symposium on Monte Carlo Methods," J. Roy. Stat. Soc., B 16 (1954), 23-75.

18. Symposium on Monte Carlo Methods (Herbert A. Meyer, ed.). New York, N.Y.: Wiley and Sons, 1956. Pp. 382.

19. The Institute of Management Science Bulletin, Vol. 5, No. 2 (1958), p.3.

20. Thomas Clayton J., and Deemer, Walter L., Jr. "The Role of Operational Gaming in Operations Research," Operations Research,5 (1957),1-27.

MANAGEMENT GAMES AND COMPUTERS

By

Joel M. Kibbee
Director of Education
Remington Rand Univac
Division of Sperry Rand Corporation

Management Games

Management games, although a relatively new educational technique, are being widely utilized, and much discussed. They are primarily of concern to the educator and to the research scientist, but since many of these games are played with the aid of an electronic computer, they should be of interest to computer people in general. In addition to the use of a computer for existing games, new games are being developed and will require programming. Many papers have been published on the educational aspects of management games; this paper has been written primarily to arouse interest in them as a computer application.

The first management game to become widely known was one developed by the American Management Association in 1956. It continues to be used, along with four or five other games since developed by A.M.A., as part of their management education courses and seminars. Over one hundred different management games now in use are listed in a forthcoming book on the subject*, and more than 30,000 executives have participated in at least one of them.

It seems worth-while to give a brief description of a typical game play for those who have never participated. The Remington Rand Univac Marketing Management Simulation will be used as an example.

The game session begins with a briefing. At this time the instructor describes to the participants the type of company they are about to manage, the economic environment, the general nature of the product, and the competitive forces they will face. He also discusses the scope of their authority, the functions to be filled, the decisions to be made, and the information they will receive.

The participants are divided into management teams, and after the briefing, the various teams meet to develop an organization, set objectives, and decide upon policies and procedures. In a typical game, involving perhaps forty to fifty executives, there might be six teams each with seven or eight members.

Games are played in "periods," with a period, depending on the particular game, being a simulated day, week, month, quarter or year. The Univac Marketing Game takes place in months. The participants are given operating statements for December and begin by making decisions for January. In addition to the operating statements they are also provided with a case history, sales forecast, and data on material and operating costs, production facilities, and shipping times.

The decisions for January are processed by the computer and operating reports are produced, and are returned to the participants. Decisions are now made for February, and so forth, perhaps for one simulated year. In a typical play the companies have a half-hour in which to make decisions, and reports are returned about ten minutes after the decisions have been submitted.

In the Univac Marketing Game, each company manufactures one product and markets it in three different regions, East, West, and South. All companies are competing in the same consumer market. The managers set price, spend money on advertising, hire or fire salesmen, set the salesmen's compensation rate, set production level, engage in special market research projects, etc. The total market is shared among the companies according to their pricing and advertising policies, according to the number of salesmen and their degree of training, and so forth. The operating reports show the sales obtained, the net profit achieved, inventory on hand, etc. The report also shows the number of salesmen on hand; companies can pirate experienced salesmen away from one another.

Each management attempts to achieve the largest possible accumulated net profit, and a "winner" might be proclaimed. This is usually discouraged, however, as good performance in a management game as in real business, depends on many factors such as return on investment, share of market, personnel policy, and numerous others that contribute to success. At the end of the game play, a discussion session takes place. This "critique" is held to focus attention on the

*Management Games, by Joel M. Kibbee, Clifford J. Craft, and Burt Nanus, soon to be published by the Reinhold Publishing Company.

lessons which were to be taught, and it gives the participants the opportunity to review their performance, discuss management principles with other members of the group, and receive feedback from the game administrator and other observers.

Existing management games vary widely in the types of models used. The original A.M.A. Game, and similar games developed by IBM, UCLA, Pillsbury and other organizations, are concerned with general management. The Carnegie Tech Management Game is based on the detergent industry. Other games exist for banking, petroleum, telephone exchanges, insurance companies and super markets. Some games concentrate on a particular management function, such as marketing, materials management or manufacturing. There is a game concerned with the management of a gas station, and three different existing games are based on an auto dealer model. The military have, of course, been playing war games for many centuries, and they are now utilizing computers for this purpose.

Most games are played by one or more management teams, where a team might be made up of anywhere from one to twenty executives. Since most games contain some obvious measure of performance there is always a "rivalry" between teams. There may or may not be a direct "interaction." In a marketing game various teams may be in competition for a common market, and the action of one team, say price of its product, will affect all other teams. In an inventory control game, on the other hand, each team is attempting to achieve the best performance beginning from the same conditions. A game with interaction is like tennis, a game without interaction is like golf. Both such games engender rivalry.

The word "competitive" has often been used in the classification of games, usually as synonymous with interaction. There is, however, an economic meaning for "competition," namely, competing for a share of a common market. A marketing game would include competition, but such competition can be either of an interactive form, with the competitors being other participating teams, or a non-interactive form, in which the economic competition is built into the model itself.

Games also differ as to the level of management for which they are intended. They differ widely as to the complexity of the model. Some are meant to be played quickly, others require considerable analysis. In general, then, a management game is a dynamic case history in which the participants, faced with a simulated business situation, make decisions, and are fed back reports based upon these decisions.

Manual and Computer Games

Management games, like the business situation they simulate, require that information be processed and calculations made. The extent of the computations depend on the particular game, and may require anything from a pencil to a large computer. This has given rise to an obvious classification into manual games and computer games. Computers are used for management games for the same reasons they are used in any business application, primarily to perform computations speedily, accurately, and automatically. Computer games can also be more flexible, as will be discussed more fully below.

Some rather odd advantages have been attributed to manual games. It has been stated, for example, that manual games can be made very simple and easy to play. Obviously a computer game can be made just as simple, though there is a temptation to use the full capacity of the computer with a resultant complexity that is not needed to satisfy the educational objectives. Similarly, one published statement claims that an advantage of manual games is less time pressure on the participants. Just because the computations are made rapidly does not mean that the participants must make their decisions quickly. Manual games do have advantages. They usually cost less initially, do not require special facilities, and can be scheduled as desired. Good design is required to keep down the number of clerks and administrators needed. There are so many advantages to computer games, however, that, at least for this author, they seem well worth the development and operating costs.

Since games vary widely in complexity, it is not possible to state how much computer time or programming is generally involved. However, several moderately complex games which the author helped develop, and which were designed for a one or two day management exercise, involve something of the order of 3,000 to 5,000 instructions, and took from three to nine man-months to program. The total development cost of a game includes more than programming, however. More or less time can be spent on the creation of the model, consulting fees can be required, as well as the cost of materials and test plays. Most often the programmer is involved in the development of the model as well as the computer coding. These moderately complex games are usually designed for a one day management exercise and might involve five to eight hours of computer time, although the computer might actually be used only part of the time and be free for other processing between game periods. As an example, a play of the Univac Marketing Management Simulation, lasting for perhaps eight to ten simulated months, and accommodating about fifty executives, might cost \$300 to \$500 for computing time. The cost is about the same if the game is played discontinuously using a Univac Service Center. However, a group playing a game might also have non-computer expenses for special facilities, materials or staff.

It is not necessary to have a computer on the premises to conduct a management game. Recently, five different cities in the Midwest simultaneously played the Univac Marketing Game through the use of leased telephone lines. Of more interest, however, is what we call the "discontinuous" mode of play. Decisions are made perhaps once a week and mailed to a service center and the reports are mailed back. This enables the game to be played without infringing upon regular production time since the game can be run at any odd 15 minutes at the convenience of the center. Many educational, industrial and professional organizations are at this moment engaged in discontinuous plays of management games.

Management Games And Computer Personnel

Because of the widespread use of management games, and their ever increasing growth, it is likely that most computer installations will at one time or another find themselves involved in running a game session. Several computer manufacturers have developed management games and happily supply the programs to their users. Most computer games developed by other business or educational organizations are also available. Directors of Training or of Management Development are now generally interested in this new tool and will probably be getting in touch with the manager of the computer installation if they have not already done so.

The computer installation may also be called upon to help develop and program a new management game. Irrespective of their interest in education, computer personnel will find that management games can provide an excellent orientation to data processing for top management. It is sometimes difficult to get a company president to watch a payroll demonstration, or a matrix being inverted, but, as personal experience has shown, he can become very much interested in the computer as a result of his involvement in a management game.

The data processing manager might consider the use of management games for training personnel within his own department. A game called SMART for systems and procedures managers was developed a few years ago, and the System Development Corporation developed a game called STEP for use in training programming supervisors.

The Educational Advantages Of Management Games

Very little research has been done on the validity of management games as an educational tool, but a similar statement can probably be made about most educational techniques now in use. The most striking thing about a management game is the involvement on the part of the participants. One is continually impressed with the way in which executives will work "after hours" in planning the operations for their simulated companies, and it is generally necessary to bring in sandwiches and coffee rather than to attempt to interrupt the play for a luncheon or dinner. Motivation is an important aspect of learning, and management games are sometimes used at the start of a course or seminar merely to stimulate "students" towards a greater receptivity for lectures or other types of training that will follow.

Management games are superior to other educational techniques for demonstrating the importance of planned, critically timed decisions; the necessity of flexible organized effort; and the significance of reaching a dynamic balance between interacting managerial functions. They can also demonstrate the need for decision-assisting tools, such as forecasts, control charts and budgets. They can demonstrate to management the power of a scientific approach to decision making.

Management games are educational tools, and to be effective should be used along with other techniques as part of an overall course or seminar. The briefing and the critique are as important as the actual play. Furthermore, most game sessions involve many "incidents" which are not actually part of the computer model. The participants might be asked to formulate a personnel policy, or to submit special reports. Job rotation, promotions, appraisals, and so forth can all be made part of the exercise. Much of the activity that takes place -- in organizing, planning, communicating and controlling -- is in addition to the numerical decisions which are submitted to the computer. Management games continue to be demonstrated for a variety of reasons, but a demonstration is not a course, and a participant should not form his opinion as to their educational merit from a few hours engaged in a demonstration play.

Building A Management Game

Management games are constructed for educational purposes, and their construction is premised on a set of educational objectives. Working from the objectives, a model which simulates a business situation is constructed. One of the most important constraints on the model is a need for simplicity.

A game must be simple to play. This does not mean that it needs to be easy to make good decisions, but the participants should not have to devote considerable time and energy to learning the rules. It requires skill and experience to abstract from the real world those elements of major importance so that a playable game will result. It is here that a programmer must work closely with the team that is building a game, and vice versa, the team should get the programmer into the act as early as possible. Skillful programming can do much to simplify the mechanics of play for the participants. A good program facilitates the manner in which the players submit their decisions, and attempts to set up procedures which will keep clerical errors to a minimum, and even possibly have the computer edit the decisions for obvious errors.

In the Univac Manufacturing Game, for example, there are quite a few shipping decisions. Suppose a company decides to ship 100 Clanks to the Western Region, and 100 Clanks to the Eastern

Region, but only has 160 Clanks on hand. The computer automatically interprets the decision as a desire to ship equally to both regions and 80 are sent, with no interruption in the game session. Similarly in games which have a limited cash on hand, the computer can issue emergency loans at some interest rate to accommodate an error on the part of the participants in spending too much money.

In some games, it is only necessary for the participants to circle code numbers on a decision form, rather than write out quantities with possible errors in the number of trailing zeros. The object of games is seldom to teach the participants to be less careless in their clerical tasks, and a good computer program can do much to eliminate clerical chores entirely so that the teams can concentrate on their decision problems.

While in university courses, and often in military games, the time spent on the game session might be lengthy, in most business applications only a day or so of a course or seminar can be devoted to the game exercise and simplicity for the participant is extremely important. This is one of the main constraints on the model, and even more on the computer program.

Experience has shown that the mathematical model used in management games can be extremely simple. One may begin with a curve, perhaps relating a demand to advertising expenditure, which is defined by a table of fifteen points. Later it is found that if only five points are used, the play from the standpoint of the participants is identical, and the author has actually used games in which all relationships are linear -- though there are many interacting ones -- without any apparent difference in the training experience. In the usually short time a company has to make its decisions, the mere fact that demand might depend on six or seven marketing decisions presents adequate complexity without the use of complex curves.

From the standpoint of the computer computations, it is not usually important whether a more or less complex curve is used, though extreme mathematical complexity could lengthen the computation time even in a large computer. However, throughout the complete model a surprisingly amount of simplification can be introduced without violating the educational objectives. Simplicity is important for the game administrator as well as the participants, since it should not be necessary to have a large staff to conduct game sessions.

As one continues to emphasize simplicity,the question of realism always arises. Some games are used to teach a specific skill or technique, perhaps production planning and control. In such cases a realistic and adequately complex model might be necessary. But most management games are used to teach general principles, for instance the importance of planning and control rather than the specific relationship between inventory carrying costs and stockout costs. For such games verisimilitude and not realism is the most important attribute.

Verisimilitude is the appearance of reality. It is as important in management games as in the theater. As long as the relationship between price and demand, for example, seems similar to what goes on in the real world, and sufficiently engrosses the participant in the exercise, it is not important that the actual curve used, assuming even that it was known, is identical with that obtained from a detailed study of real data. Usually one is attempting to train a manager in, for example, marketing principles, and not the way in which a particular product with which he is concerned will behave in the real market.

The word "Management Game" has generally been used with a training implication, and the term "Simulation" used when the object is to solve a problem, or actually guide management decision making. Under this terminology a simulation model must have validity, in the sense of an ability to predict the future, if it is to be of value; the management game model, used for training, must more often stand the test of verisimilitude. In fact, it is possible that an over concern with realism can produce a game that is too complex, too difficult to play, and can actually destroy verisimilitude, and the involvement on the part of the participants which is so important.

The programmer working together with a team that is constructing a management game can do much in the cause of simplicity and verisimilitude. It is this very ability which makes computer games, in general superior to manual games. The reports returned to the participants can resemble the actual reports which they obtain in everyday life. Furthermore, there is essentially no limit to the amount of special information that can be provided to simplify the role of the participant. Thus, a report can give total costs, unit costs, and percentages. It can provide a variety of research type information and statistics. Little of this is usually possible in a manual game. The American Management Association's Top Management Decision Making Simulation permits management to engage in market research studies, at a suitable cost, which will provide them with the answer to questions concerning the number of orders for a product that might have been received had a different price been set. From the computer standpoint, it is only necessary to loop through the same set of computations using the research price rather than the actual price.

A good computer program can also simplify the task of the game administrator. In the American Management Association's General Management Simulation, it is possible during the play to develop additional products through a program of research and development. When a product has been developed the staff sends a pre-printed letter to the company informing them of this. However, the computer program has been arranged so that a special report to the administrator is prepared each period, and the computer itself, on this report, informs the administrator when to send out a particular letter. This same report, incidentally, informs the administrator about various aspects of the company's performance so that he is better able to control the session, and to provide feedback at the critique.

There are many ways in which the relationship between quantities can be introduced into the model. The most common approach is to use a mathematical function; this itself may take the form of a graph, a table, or an algebraic expression. Another approach is the use of judges. There might be one or more "expert" judges who make a subjective evaluation of the effect of particular policies. Such a judge should not be confused with a clerk, sometimes called a judge, umpire or referee, who, in a manual game, merely performs the arithmetic computations according to pre-arranged formulas. This is the function of the "computer," whether it be a man or a machine. The "expert" judge is a person with specific knowledge and experience, who arrives at a subjective evaluation of the decisions, hopefully without bias towards the particular participants.

The judges themselves may be a group of participants, and can be considered to be one of the playing teams. One might have five competing companies in a general management game, together with two teams of competing bankers who may invest money in the various companies. A model may use various combinations of equations, judges, etc., and in the example just cited only the capitalization aspect is relegated to judges. In some games only certain non-quantifiable factors may be left to observers who can directly influence company performance in the role of judges.

In one game based on a real product, the judges consisted of several members of top management, and their own deliberations and arguments as to the evaluation to be placed on the various marketing policies being exhibited by the teams playing the game proved to be, in their own opinion, an extremely valuable educational experience. At the other extreme, one might use judges who did not even know that a game was being played. For example, marketing policies together with advertising copy could be presented to an external group of simulated "customers," for their own preference in the products being offered.

A very objective and unbiased manner for incorporating the necessary relationships in a model is by the use of bidding techniques. For example, each company might submit a closed bid as to how many units they could supply at a particular price, and the demand would be awarded on the basis of lowest price. Similarly, raw material could be offered to the highest bidder, and the cost would then not be based on a specific value built into the model. Bidding techniques have been highly successful in many games. A particularly good example is the "Management Business Game" produced by the Avalon Hill Company, an excellent "parlor" game that can have serious uses.

Parameters

The mathematical meaning of parameter applies mainly to the constants used in the algebraic relationships between quantities, but for games it is convenient to extend this concept to all constants. For instance, the cost of carrying inventory in a game might be a function of the opening value of the inventory A and the closing value of inventory B, namely, .05 (rA + sB). The .05 is also a parameter, but it has been given a specific value for this illustration. If we set $r = 1$ and $s = 0$, we have a cost dependent only on opening values. We might prefer to set $r = \frac{1}{2}$ and $s = \frac{1}{2}$, which is slightly more realistic. Thus, we are changing the nature of the model by our choice for r and s. Similarly, the cost of raw material is a parameter which may be changed for different game plays. The inclusion or omission of certain factors can be controlled by parameters acting as switches. Whether or not certain information is to be included on the report can be controlled by a parameter which can take on the values of 0 or 1. In well-designed computer games, extensive tables of such parameters are used, and in this way seemingly different games can arise from the same model by the choice of a particular parameter set.

One normally attempts to compute in advance the parameter values that will be used in a particular game, but this is usually followed by actual parameter studies once the program has been checked out, and, of greater importance, by numerous test plays. There is always a range of feasible parameter values, and the particular set to be used in a particular game session will depend on the game administrator. A game should be packaged with many such sets (and they may be labeled "highly stable somewhat price insensitive industrial product" or "highly competitive very price sensitive consumer product" and so forth.) Thus a single game program can be used for different games, and ideally one can imagine one large model with sufficient parameters to allow it to be adopted to a variety of industries and situations.

Extreme Values And Limits

Like any good program the game program must be prepared to handle any decisions, no matter how ridiculous or extreme. Many a program has "hung up" during a game session because the game designers were convinced that only a particular range of decisions might rationally be made. Not only must the program handle extreme decisions, but a rational result must be obtained. For example, a product might be priced somewhere around $10.00, but the program should be ready to accept prices from $0 to $999999999 or whatever the upper limit is that is imposed by the input design. At some suitably high price the demand will go to zero, but it must remain there, and a curve must not be used that departs from zero again above $999 just because one does not expect such a decision to be made. It must also be decided what the demand will be if a price of $0 is set, even though the company that makes it will be losing money on every item sold.

The American Management Association's General Management Simulation is immune to participants who might take extreme decisions. It is possible for a company to fire all of its workers, close down plants, etc. The program not only will handle this, but will change its overhead cost distribution procedure accordingly! As in any computer program, the programmer must be prepared to have any quantity, unless limited by input format (though that is just another method under his control) take on any value from minus infinity to plus infinity.

The Future Of Management Games

Management Games are only a few years old, but one can already look back with fondness on their infancy, and look ahead with confidence to their maturity. Like most babies, the first few games were very much alike; they usually modeled the marketing or manufacturing of a durable good and were slanted at higher management. Today new games, like new teen-age singers, are arriving on the scene with increasing rapidity.

Management games have been used primarily as an educational tool. Their use in training is increasing and will increase, and will also spread well beyond the area qualified by the word "management." In addition, they will undoubtedly have considerable application in research, problem solving, personnel testing, and as a direct aid to management decision making.

There are still no management games for mining companies, the fishing industry, or the mink farm. The entertainment field -- TV, publishing, motion picture studios -- need management as well as "talent." Government -- city, state or national -- provides a large area of application of management games for training. Labor unions, universities and professional associations also have managers. It is fairly easy to write down hundreds of training situations which could well use this new educational tool. And perhaps somebody ought to build a game to teach people how to build a game.

Games completely different from those now in use can be expected. A super-game could be constructed which included manufacturing companies, financial institutions, service organizations, suppliers of raw material, and even a couple of management consulting firms. Management games are having extensive use in management education, but there is probably an even greater need for new tools in supervisory training.

Psychologists and sociologists have long used humans, as well as animals, to study human behavior. Much work has been done with small task performing groups. The computer opens the possibility of new uses of simulation in the life sciences, and one can expect an increase in the number of laboratories now doing such research. Management games can also be expected to play an important role in economic research.

While the simple manual management game has a purpose, and is extremely useful in many training situations, one can safely predict an increasing use of computers in the management game area. This paper was presented because of increasing importance of management games to computer people. It is hoped that the interested reader will read elsewhere for those more important aspects of management games related to their construction, their educational utilization, etc. And it is also hoped that he will find the opportunity of playing one -- it is not only fun, it is educational.

AN ON-LINE MANAGEMENT SYSTEM USING ENGLISH LANGUAGE

Andrew Vazsonyi
Ramo-Wooldridge
Canoga Park, Calif.

Summary

The demonstration model of an on-line management system presented in this paper aims to provide increased management capability to executives charged with planning and control of large scale research development and production programs. The technique is formulated as an exercise in Decision Gaming and special emphasis is laid to the problem of providing capability for quick and optimum reprogramming of dollars, manpower, facilities and other resources. The task of planning and control is structured into two components, the more routine tasks are assigned to the equipment, whereas problems requiring executive judgment are delegated to players of the Decision Game. Through the use of mathematical models and computer routines the consequences of proposed reprogramming actions are presented to the players in terms of financial and manpower requirements, facilities loading, etc. Through a step by step man-machine process, optimum programs and the best utilization of resources is reached. Management data is retrieved and manipulated on an on-line basis and all operations of the equipment are executed through every day English commands. All data is displayed on cathode ray tubes and projection screens, including instructions to the players on how to operate the equipment and how to play the Decision Game. Input to the equipment is provided through (1) a permanently labeled keyboard, (2) a blind keyboard that can be provided with appropriate labels through a set of plastic overlays. The computer action resulting from depressing of keys must be programmed and is not wired permanently. By providing a set of independently operated input-output consoles, connected on-line to the same computer system, a significant advance in the art of the design of management systems is provided.

Introduction

The management planning and control technique described in this presentation has been developed for certain military and civilian activities with the purpose of assisting executives in evaluating and re-programming complex activities. However, it is believed that the technique is equally applicable to the planning and control of other large scale research, development, production and construction programs.

In order to apply the planning and control technique to an activity it is necessary to divide the activity into "elementary" programming blocks. The technique requires that first alternate scheduling and financial data on each of these planning blocks be developed. In addition, it is necessary to formulate explicitly the inter-relationships between the planning blocks. These relationships specify the permissible time phasing of the elementary programming blocks, the associated dollar and manpower requirements, facilities requirements and other financial requirements. The planning and control technique employs a network analysis of the various activities involved and permits the exploration of a large number of planning combinations.

The primary purpose of the management planning and control system is to assist executives in re-programming. As an illustration, suppose that plans are compared with progress, a deviation is observed and re-programming of the different activities is required. For instance, it might be necessary (1) to cancel a program, (2) to stretch another one out, (3) to accelerate one, or (4) to decrease or increase production quantities. Another situation when the need for re-programming arises, when there is a budgetary change and financial trade-offs between various programs must be considered. For instance, executives may want to know that if a particular deadline is postponed by six months, how many dollars and what manpower can be made available to another program, and by how much can this other program be accelerated.

The actual program analysis and re-programming activity is carried out through the medium of a Management Decision Game.

Brief Description of Decision Gaming Technique

The Decision Game is to be played in three steps. As a first step the players of the Decision Game gather in the Control Room where the Game is to be played and where the various information displays can be retrieved with the aid of the computer system. The displays present all the important planning factors relating to the activities to be re-programmed. The time phasing of various missions, deadlines and goals, and the associated loading of various facilities can all be displayed. The associated financial information can also be shown with sufficient detail so that financial consequences of re-programming decisions can be made. Provisions are made to retrieve further back-up information when requested, from a file of status of progress and alternatives.

As a first step of re-programming a comprehensive analysis of the status of the programs is carried out. The computer system is provided with the capability of furnishing status information on a real-time basis and in everyday English. Information related to all matters pertaining to the progress of various programs is displayed in cathode-ray tubes or in projection screen. After this status analysis is completed, the players have adequate information to perform the second step of the Game.

This second step of the Game consists of making a "move". Such a "move" may involve a time shift of some of the deadlines, milestones or goals and/or a change in the delivery quantities involved in the program. As suggestions for re-programming "moves" are made, the proposed changes are put into the computer system through the use of an appropriate keyboard and Communication Display Tube. At the direction of the operator the computer and associated equipment takes over and the third step of the Game, that is the re-programming computations, are carried out.

This third step of the Decision Game is executed by the computer in accordance with mathematical models and associated computer routines stored in its memory. Within a time span of seconds the computer prepares a new program, including all the new deadlines the new phasing of sub-programs, facilities loading, manpower and financial implications. When the computer finishes the computations, the data is presented to the players through cathode-ray tubes and/or slide projections. By examining the various displays and by retrieving more detailed information, the players can evaluate whether the suggested solution to the re-programming problem is satisfactory.

In most situations the first suggested program will result in conditions that are not acceptable to the players of the Game. Therefore, after considering the results of the "move" and discussing further implications of the data, a new proposal for re-programming will emerge and a new cycle of the Decision Game will be entered upon. By a series of steps it is possible to develop a final program that is acceptable to the players.

At the termination of the gaming exercise all the implications of the final program are recomputed with greater accuracy. It is not expected that this re-computation will result in major changes, but only that the re-computation will provide an accurate, acceptable and detailed plan.

All communications between man and machine are performed in a real-time manner and in everyday English.

The File of Status and Alternatives

The Gaming Technique described here allows the examination of a panorama of alternatives. The analysis can be performed only if in the memory of the computer, techniques for examining many alternate possibilities are stored and programmed. It is recognized that it is impossible to store all the possible alternatives and therefore, a method to study alternatives must be provided. The analysis is made possible by the application of mathematical modeling techniques and by the storage of certain basic system parameters. The mathematical model uses the elementary programming activities as basic building blocks and relates these activities to each other through mathematical relationships. For instance, alternate ways to accomplish a basic programming block can be associated with various estimates of completion dates and costs. The mathematical model summarizes the relationships, and also relates the different activities to each other through equations and inequalities. Manpower and financial requirements appear as dependent variables, whereas the time phasing of the various activities as independent variables.

In order to avoid the necessity of manipulating a large number of parameters, sub-optimizing techniques are introduced. For instance, it might be required that certain types of sub-programs be accomplished at a minimum cost and this policy can be embodied in a system of equations through mathematical programming techniques. By such relationships, the majority of the variables of the system can be made to depend on a few control variables. With the aid of mathematical models and sub-optimization techniques it becomes possible for the players to manipulate only a few of the major variables and still examine a large number of alternate plans.

Equipment Requirements

There is no equipment on the shelf today that can carry out in all its details the management planning and control technique described here. However, there is equipment available, which with minor modifications would possess the capability required. A detailed study of the Ramo-Wooldridge Polymorphic Computer System and Display Analysis Console, for instance, shows that essentially all the required features could be made available in a short time. This computer system has been described elsewhere,and in this discussion equipment details will not be included.

Decision Gaming

Detailed Description of Decision Gaming System

The fundamental concepts underlying the Decision Game are shown pictorially in a simplified form in Figure 1. Three displays enable the players to communicate with the computer. The first of these is a visual representation of the time phasing of all the important missions and goals. The information on this display is schematically represented in the upper part of Figure 1 and is to be displayed in the "Program Network Tube" of Figure 2 (projection capability can be provided if desired so that a group of participants can analyze the data). Sufficient details will be shown so that all milestones of importance are displayed, but the data will not be so detailed as to confuse the players. As the Game starts, various questions will arise which will not be immediately answerable by the displayed material. To meet this condition, back-up displays will be stored which can be retrieved by the players as requested. By this technique, it will be possible for the players to go into any degree of detail in the time phasing of the missions and goals without making the presentation too cumbersome or confusing.

A part of the display on the "Program Network Tube" is the visual representation of the utilization and loading of the different facilities associated with the programs considered. This display is shown by the third item from the top in Figure 1. All the previous comments made in connection with the visual representation of Programs A and B apply for the Facilities Loading displays, too. Sufficient detail will be given so that the player can appraise the state and progress of various programs, and again sufficient back-up information will be available through retrieval.

The second display refers to dollars, costs, manpower, and other resources. These are represented graphically in the lower part of Figure 1 and are to be displayed on the "Resources Requirements Tube" of Figure 2. The dollar and manpower profiles as they unfold in time will be represented in sufficient detail so that all the important information for the players will be furnished. In addition, when it is required, the players will be furnished with hard copies of printed financial information.

The display capability so far described furnishes the players of the Game with such pertinent information as past history, status, and future projections of programs. Particular emphasis is placed on the preparation of this information in such a form that organizational structure and responsibilities are directly tied in to the information displayed.

The lower right corner of Figure 2 shows the "Man Machine Communication Display". This is the tube that offers choices of instructions to the player in plain English. This tube is used mostly for non-standard type of instruction to the player, as ordinary instructions (say: "Machine Is Busy") are provided through the illumination of status lights.

So far we have described the display systems and the type of information stored. We are now ready to proceed to the description of how the Decision Game is to be played. In order to be able to speak in more specific terms, we take the hypothetical problem of a new requirement, that a particular mission is to be accomplished one year ahead of schedule. This new requirement requires the acceleration of a major program and a reorientation of the resources available.

When such a problem arises, various discussions take place at different managerial levels. We do not propose that the Decision Game is to replace these conferences. However, after a preliminary consideration of the problem the appropriate management group gathers in the control room to play the Decision Game. By a step by step procedure, they evaluate, modify and sharpen the preliminary ideas that have risen in connection with this problem of advancing the completion date of a major mission.

When the group meets the first time in the control room, the players begin by retrieving a number of different displays to update and verify their knowledge of the status programs. Such a review consists of inspecting the principal displays associated with the problem and of retrieving various back-up information. After such a preliminary discussion, a proposed first solution to the reprogramming problem is suggested and information defining the proposed change is keypunched into the computer.

At the instruction of the players the computer begins to carry out the routine associated with the particular reprogramming problem introduced. The computer consults the Data and Program Reservoir containing the file of status and alternatives shown on the lower left-hand side in Figure 2, and on the basis of stored information and routines, computes dollar and manpower requirements. In addition, facilities requirements and loading are checked and computations are made to determine whether the desired acceleration is feasible at all.

As the computer proceeds through its routine, it might find that the proposed acceleration is impossible or impractical. It is possible that even if all projects are put on a crash basis the mission could not be accomplished within an acceptable date. It may be that for instance manpower is not available, even if more shifts are employed. Under such conditions, the computer will indicate that the plan is not feasible and it will display on the "Communication Display Tube" a warning signal, which shows in detail why the proposed solution to the reprogramming problem is not feasible.

At this point, a group discussion follows to determine whether by a higher order of decision a solution could be found. For instance, it might be decided that another facility can be built or made available, or that another contractor can be called in. Information available to the decision maker will not always be programmed into the computer and, consequently, feasibility indicated by the computer will occasionally be considered as tentative.

If, indeed, a need for such a new alternative way of proceeding with the problem exists, this information must be put into quantitative form and fed into the machine. On the other hand, if the computer indicates general feasibility, then the players can immediately proceed to further evaluation of the proposed program.

When the program modification is feasible, the players are primarily concerned with resource requirements and with dollar and manpower profiles associated with the program. It is very likely that the first solution proposed will not be acceptable from the point of view of budgetary considerations. It is likely that the costs at certain phases of the program will be beyond possible funding, and perhaps at some other times there will be an indication of surplus funds. This, then, is the point where the players reconsider the time phasing of the mission and goals and propose an alternative. When the players agree on the next trial of the program phasing, information is fed into the computer and the computer proceeds with computations to prepare a new program. Again, the computer first explores feasibility and then proceeds to the detailed generation of the resource requirements.

It is seen that through a step by step process of deliberation, discussion and computer routines, the players will reach better and better solutions to the reprogramming problem. It is envisioned that programming computations will be carried out first by a "quick and dirty" method and then by a more accurate routine. This will allow the players to explore tentative alternatives rapidly and there will be no unnecessary delay in waiting for accurate computations which would not be utilized in actual program plans. The computer will carry out accurate computations either automatically (when computing time is available) or at the special direction of the player. This approach allows the decision makers to make rapid changes and explore and evaluate dozens of different program proposals. As the Decision Game progresses, more and more satisfactory solutions to the reprogramming problem will be found. Towards the terminal phase of the gaming exercise, the players may desire highly accurate estimates of the various program details. If this is so, it may be necessary to direct the computer to carry out more accurate special program computations, and it may then be necessary for the players to wait for a longer period of time to get the phasing of programs and the resource requirements. Finally, the computer is directed to develop and print a definitive program which will be used as a planning document. Computation of such a program may require hours, and consultation with other agencies and contractors.

So far, we have given only an outline of how the Decision Game is to be played and described only those phenomena that will be observed by the players. Now we proceed to take a look inside the equipment and see how the various logical steps, routines and computations are carried out.

Illustration of Reprogramming Computations

The basic principle in carrying out reprogramming operations is to provide the computer with data on possible alternatives and also with the myriads of details on how these alternatives can be combined into programs. The computer can be programmed to go through a large number of calculations in an efficient fashion, and therefore alternate programs can be generated by the computer in a matter of seconds. In order to illustrate the techniques, we will describe an extremely simple but still significant reprogramming problem.

Figure 3 is a chart showing six different jobs and the time phasing of the start and completion dates of each of these jobs. In this simplified programming Game, we are concerned only with the monthly dollar expenditures which are shown in the bottom of Figure 3. Suppose the player desires (1) to accelerate by two months the accomplishment of Goal B (that is the terminal dates of Job No. 3 and 5); (2) to accelerate by three months the final completion of the mission, that is of Goal A; (3) leave all other goals unchanged. The computer is to determine whether such an acceleration in the program is feasible, and what kind of dollar expenditures would be associated with this accelerated program.

As this reprogramming information is keyed into the machine, the machine examines all jobs to see which is immediately affected by the acceleration of Goals A and B. The computer selects Jobs 3, 5 and 6 and evaluates the possibility of accelerating those three jobs. It finds that the time span of Jobs 3 and 5 are to be compressed by two months and of Job 6 by one month.

At this point, the computer seeks information on alternative ways of accomplishing Jobs 3, 5, and 6. As the computer consults the file of alternatives, it finds for each job the time-cost relationship shown in Figure 4. The horizontal axis shows alternative time spans allowed for the job, the vertical axis shows the total dollars that must be expended, if the job is to be accomplished in the time specified. It is seen, for instance, that a crash program--doing the job in the shortest possible time--requires more total funds than a more orderly and efficient execution of the task. In the case of a stretch-out, due to overhead and some other supporting activities, the total cost of the job would also increase. The computer also finds how these dollars would be expended in time. (Dotted lines in Figure 4.) The file of alternatives has curves of this type for each of the jobs and therefore the computer can establish that the jobs can indeed be accelerated to the desired time span, but that a higher expenditure of funds is required. Using this information, the computer can replace the previous budgets for Jobs 3, 5, and 6 with the new budgets and determine a new dollar profile associated with the accelerated program. We see that when the computer reprograms, it first proceeds through these computational steps and then transmits the information to the display devices. The player can visually observe the required funding associated with the accelerated program.

We recognize that in a real problem we would deal with a much more complicated set of routines. Manpower profiles would have to be computed, facilities loadings would have to be checked, many other items of information on compatibility would have to be considered. In the case of prototype production, or in other tasks where quantities are involved, relationships dealing with "quantity made" would have to be included in the analysis. However, basically, these considerations would only complicate (admittedly by a great extent) the routines that the computer would have to go through, but, conceptually, reality would not add significant new difficulties to the method of solution.

The time cost relationships as shown in Figure 4 form the basis of the file of alternatives that a computer has to consult. As we already mentioned, there are types of problems where more complex mathematical models form the building blocks for the file of alternatives. However, for purposes of our discussion, we will concentrate on the concept of time-cost relationships and we will show how such relationships can be generated. We will show how the basic input data is to be obtained and how these data can be built into the appropriate files for representing various alternatives that the programming task may require.

Concept of Alternatives

Let us reiterate the type of information we seek. The player moves some of the gaming goals in time and certain jobs must be performed within the time limits indicated by the player. We need to find a way to determine the dollar requirements associated with the various alternatives.

Let us begin by considering a relatively simple job or task. Suppose that there is a single manager in charge, and let us assume that this manager has a good grasp of all the details involved of this particular task. The manager does his own planning with paper and pencil and by discussions with his associates. We ask him to determine how much would it cost to perform this job in an "orderly" fashion. After studying the problem, he estimates manpower, material, overhead and dollar requirements. In Figure 5, the financial information is shown in a graphical form. In the horizontal axis we show the time allowed to complete the task; on the vertical axis, we show the associated effort (say dollars per week) required. "Orderly" performance of the task is represented by the "most efficient" point in the chart. We also ask the manager to determine what it would take to complete the job on a crash basis. He would need more men, more resources, he would require a larger effort, but he could complete the job in a shorter time. This crash program is shown in our chart in Figure 5 by the "minimum time" point. We can also ask him to determine the minimum level of effort required to do the job at all. He needs two mechanical engineers, an electronic expert, a technician, a secretary. This establishes his minimum effort level and gives the "minimum effort" point in Figure 5. We connect the three points by a curve and obtain a time-effort relationship and we assume that we could also operate at intermediate points on this curve. With the aid of the curve shown in Figure 5, we can determine the time-cost relationship shown in Figure 4. All we have to do is to multiply the rate of effort by the time required for the job, to get total costs.

We see, then, that we have a technique to get time-cost relationships, at least for relatively simple jobs. However, if we want to extend this technique to more complex tasks, we run into problems. It is difficult or impossible to find managers who have all the details of a complex job. Consequently, in order to make cost estimates, the manager must work with his subordinates and must combine in a complex fashion many items of information. This combination of data is a tedious and difficult job but is precisely the kind of task that computers can execute with great efficiency. Therefore, we propose to prepare time-cost curves for complex jobs with the aid of computers. We will show how, with the aid of mathematical models and sub-optimization technique, one can construct time-cost relationships.

Sub-Optimization Considerations

Let us take a simple combination of two jobs which have to be performed in sequence. Various alternate time spans are allowed either for Job No. 1 or No. 2. This implies a number of combinations of ways that the two jobs can be performed. In Figure 6 we show the problem in a graphic way. Suppose tentatively we select a certain duration for Job No. 1, and we determine the associated dollars required with the aid of the time-cost relationship. In Figure 6 this time-cost relationship is represented by point A. Now by starting with this time span, we can assign different time spans to Job No. 2. A possible representation for Job No. 2 is point B. It is seen that we can combine the two time-cost curves in many different ways. In Figure 7, the various possible time-cost curves for Job No. 2 are shown by dotted lines. Now we need a policy to select, out of these many possibilities, the desirable ones.

Suppose we agree that we want to complete the two jobs within a given time span, but with the least amount of money. Let us recognize that when the combined time-span for the two jobs is specified, still there are many ways to do the two jobs; out of these many possibilities there is one that yields the lowest cost. In Figure 7, these low cost combinations are represented by the envelope of the dotted curves. We say then that this envelope, corresponds to our policy of minimum cost, and this envelope is the combined time-cost relationship for the two jobs to be performed. For instance, if we wish to complete the two jobs at point P in Figure 7, we draw the vertical line from point P until we reach the envelope at point Q. This gives the combined cost of the two jobs. Working backwards from point Q, we can get point R which represents the time and cost requirements of Job No. 1.

The policy we used here is to perform the two jobs with the lowest possible cost. If there is another policy such as say a constant manpower requirement or the utilization of a facility, etc., each of these policies would have to be programmed into the computer. The important point, however, is that even if complex policies are formulated, due to the high-speed capability of the computers, consequences of these policies can be deduced efficiently.

Actually, the computer would not construct the envelope of the curves, but would solve the appropriate mathematical problem. It is easy to show that the two jobs are to be combined in such a fashion that the following equation holds:

$$\frac{dC_1}{dT_1} = \frac{dC_2}{dT_2} \qquad (1)$$

Here on the left-hand side we have the derivatives of the time-cost relationship for the first task and on the right-hand side, the derivative relationship for the second task.

The computer would compute these derivatives, select the appropriate combinations of the tasks and generate the new time-cost relationships.

In Figure 8, we show a somewhat more complicated problem when a sequence of jobs is to be performed. Here it can be shown that the following equation must hold:

$$\frac{dC_i}{dT_i} = \lambda \qquad (2)$$

The meaning of these equations is that the derivatives of (that is the slopes to) the time-cost curve must be equated. This procedure can be observed in Figure 8 by considering the three upper curves and realizing that the three tangents shown are all parallel. Another representation of the same set of equations is shown by the lower set of curves. These are the derivatives (or slopes) of the time-cost curves. The corresponding points on the time-cost curves are selected by taking points on the same vertical level. Again, this is the type of computation that a computer can carry out very efficiently.

Another way to describe the technique used here is to realize that whereas many goals are to be manipulated during the course of a Decision Game, some of these goals are not sufficiently important to be manipulated directly by the players. Therefore, some "slave" goals are automatically manipulated by the computer. We can say that placing of the slave goals is accomplished by an appropriate sub-optimization technique. For instance, in the discussion so far, we sub-optimized by using least-cost job combinations. As stated before, some other principle might be involved in positioning of the slave goals and then other corresponding sub-optimization principles must be developed. It is also possible that in some complex situations, one would have to be satisfied by accepting a relatively "good" solution instead of trying to find a sub-optimum.

In Figure 9, we show a somewhat more complicated problem. Those goals marked with crosses can be made slave goals by the technique so far described. However, goals A and B are interconnected as they have to be completed at the same time, and therefore this interconnection must appear somehow in the computational procedure. What we have to do is to take the time-cost relationship for the first and second jobs up to A and B, add the cost of these two jobs together and construct a single time-cost relationship. We have to go through the same procedure for the jobs to be performed after B, and form a single time-cost relationship. When we have these two time-cost relationships, we have reduced our programming problem to the problem of having two jobs to be performed in sequence. Now we can use the technique already developed.

In a way we could say that first, we turn into slave goals those goals which are in series, and then those which are in parallel. By use of this principle step by step, we can construct the necessary time-cost relationships for complex programs.

In summary, we can say that we get basic data on relatively simple jobs from managers of simple projects. Then we formulate the rules of combining these simple jobs into complex jobs, and through some method of selecting the most appropriate combination, we construct combined time-cost relationships.

Let us, however, recognize that when we deal with really complex structures, it might not be possible to put into logical or mathematical form the policies that yield the most desirable combination. If this be the case, it is necessary to resort to auxiliary gaming technique to establish the file of alternatives.

The problem shown in Figure 9 would be solved now by a group of executives moving goals A and B and by examining the consequences of these moves. Here we have to perform the same type of gaming as we have previously described. In Figure 10 we show in a schematic form, the multi-state man-machine gaming system that we envision here. On the top we show the game that we have already described and which is to be played by top executives. On the lower level we show subsidiary games which would be performed by middle management personnel. The purpose of the lower level of gaming is to provide a planning and control system for middle level management and to provide the files of alternatives to top management.

Perhaps the most significant aspect of this multi-stage gaming technique is that various levels of management could participate in a most effective fashion in reprogramming efforts. As problems develop at lower levels of management, these are reviewed by middle management and the implications of changes in program phasing are incorporated into plans. Even more significant is that not only single plans are developed, but alternative possibilities of tackling jobs are considered. When middle management agrees on various alternatives, these are placed in the file of alternatives and thereby these alternatives are made available to top management. This way top management is apprised of the most recent and significant changes in the time phasing of programs and is provided with a capability of using the best updated information.

Confidence Factors in Programming and Scheduling

We have so far attempted to divide the planning task between equipment and man in a systematic way. We recognize that a great many logical and mathematical tasks must be performed in order to generate program plans, and that many of these tasks can be performed better by computers than men. We believe that the capability of computers surpasses human judgment in one more specific area, namely in connection with the problem of estimating the degree of uncertainty associated with estimates of dates of computations of various tasks.

It has been found that human judgment is fairly good in estimating upper and lower limits of when a job will be completed, provided the task to be performed is relatively simple, and provided the man who makes this judgment is completely familiar with the job to be performed. However, when people combine the various component estimates of complex jobs, we find that it is difficult to get reliable answers.

We show in Figure 11 the problem in a highly simplified form. Suppose there are three tasks to be performed in sequence and for each, there is an uncertainty of the completion date. These uncertainties are shown in the diagram by the shaded areas, lower estimates being the optimistic ones while the higher ones are the more pessimistic estimates. In the lower part of the diagram we add the times required for the three jobs together and also add the uncertainties (three shaded areas) into a single one. How to measure now the uncertainty in completing the three jobs?

The total variability is of course shown by the sum of the shaded areas. However, it is unlikely that all three jobs will be completed at the earliest possible completion date, or conversely, that all three jobs will take the longest time estimates. Therefore, we can say that whereas the total variability is shown by the combined shaded area, the area does contain some unrealistic completion dates.

If we think in terms of a more complex program, our problem becomes more acute. When there are hundreds of jobs to be performed, it is impossible for the unaided human brain to form a composite picture of the probabilities involved.

However, this is a sort of problem that statisticians have already studied. In Figure 12, we show a simple example when three different jobs are to be cascaded. If we estimate probability distributions of completion dates for each task and associated standard deviations, then at least under certain simplified conditions, we can use the following equation for determining the standard deviation of the composite probability distribution:

$$\sigma^2 = \sigma_1^2 + \sigma_2^2 + \sigma_3^3 \qquad (3)$$

In this equation on the right-hand side are the squares of the individual standard deviations, while on the left-hand side is the square of the composite standard deviation.

So far, we have talked only about a very simple situation. However, recently some mathematical studies have been made, on how to combine probability distributions for complex programs. It is believed that when these techniques are combined with machine computations, one can obtain more reliable estimates of completion dates than is possible today by unaided human judgment.

Weapons Systems Programming and Control System

(An Illustration)

The United States Air Force controls one of the largest and most complex research, development, production and operational programs in the world. Management of these programs represents a formidable task and considerable effort is devoted to develop new and better management techniques. One of these management planning efforts goes under the name of WSPACS, or Weapons Systems Programming and Control System.[1] The objective of this system is to provide the Air Force with a broad planning device and also to provide techniques of use to both Air Force and Industry in maintaining control and surveillance over the expenditures of development and production contracts. In order to demonstrate that a man-machine management system will furnish the required capability, a demonstration model has been recently constructed and tested. So far this WSPACS Mod 0 model has been programmed on a conventional computer, but here in this discussion we want to explore the possibility of how such a weapons system programming and control system could be carried out within the framework of our on-line management system. As a method of presentation we will use the description of a hypothetical exercise in reprogramming.

Let us say that at a certain day the planning staff of the appropriate Air Force organization is called together and is advised that a change in the planning of programs is required. Specifically, the problem arises from two new requirements. (1) There is a reduction in next year's fiscal expenditures from $4. 6 billions to $4. 4 billions; (2) it becomes extremely desirable to accelerate the Air Force's missile programs.

When this problem of reprogramming is presented to the staff they gather around the display console of the computer system and begin an analysis of the various Air Force programs. The man-machine system employed is shown in general, in Figure 2, the special keyboard overlay to be used in the reprogramming exercise is shown in Figure 13. The key labeled "Start Routine" is lit (from under) indicating that this is the key the operator must depress to start the analysis. As soon as this key is depressed the following instruction appears on the Man-Machine Communication tube:

```
THIS IS A WSPACS ANALYSIS. CONSULT
YOUR MANUAL BEFORE PROCEEDING. IN
ORDER TO CARRY OUT ANALYSIS DEPRESS:
"PROCEED WITH ROUTINE" KEY.
```

The operator inspects the keyboard and realizes that in fact only the "Proceed with Routine" key is lit and that therefore this is the only key he is permitted to depress. He proceeds to depress this key.

On the "Resources Requirements Tube" of Figure 2, (on the right hand tube) the information shown in Figure 14 is displayed. The staff observes the various Air Force programs and associated financial information. For each program D and P, that is design and production and Sys-conn, or systems connected expenditures are shown. On the bottom Non-System costs, total Expenditures and expenditure Limitations are shown. Let us realize that in this display not all weapon systems are shown and that financial information is shown only up to 1965.

However, by inspecting the lower left-hand portion of the keyboard, we notice that provision is made to scan tables of information. For instance, if the key "Right Tube" is depressed and simultaneously the key labeled by the arrow pointing to the left is depressed, then the numbers shown in the columns in the right-hand tube will shift to the left and financial information for 1966, 67, etc., appears. By this means, any limitation on the horizontal and vertical capacities of the cathode-ray tubes is overcome.

The staff now analyzes the financial data shown on the right-hand tube and decides to proceed with the analysis. They note that the Man-Machine Communication tube is displaying the following statement:

```
YOU MAY SELECT PROGRAM FOR ANALYSIS
ON ALPHA-NUMERIC KEYBOARD.
```

The operator also notices on the keyboard that the key "Operator: Select on Alpha-Numeric Keyboard" is lit. (This is further verification to the operator that he is to use the Alpha-Numeric keyboard.) The staff decides to proceed with an analysis of the Atlas Program, therefore the number 1 is key punched on the Alpha-Numeric keyboard. At this instant on the Man-Machine Communication tube the following statement appears:

```
YOU SELECTED ATLAS FOR ANALYSIS.
YOU MAY INTRODUCE THE FIRST
ALTERNATE IN ATLAS PROGRAM BY
DEPRESSING "PROCEED WITH
ROUTINE".
```

Simultaneously on the Program Network Tube (to which we refer to as the left tube), details of the Atlas program appear. It is noted that there are in total 276 units in the program, that there are 12 units per squadron, that so far 8 units have been delivered and that there are no active squadrons as of today. The authorization of the Atlas (go-ahead date) was May 1958, and the last delivery date is June 1964. The left-hand tube also shows schedules and expenditures for Atlas. For instance, in 1961 there are 49 units to be delivered and 4 squadrons projected. Atlas expenditures are $65 million for design and production and $316 for systems connected costs. Total non-systems cost for all programs are $1,200,000,000, leading to a total Air Force expenditure of $4,591,000,000. (The new expenditure limitation is $4,400,000,000.)

Similar data is shown for each fiscal year up to 1970. We recognize that not all these data can be put on the tube simultaneously. However, with the aid of display control keys we have the capability of scanning these tables up, down, right, and left.

At this instant the planning staff is studying on the right-hand tube the financial aspects of all the weapons systems programs, and on the left-hand tube details of the Atlas program. With the aid of a retrieval system not described here, further information relating to Atlas and other Air Force programs is displayed and analyzed by the staff. After considerable exploration and discussion it is proposed that a trial be made to modify the Atlas program. The "Man-Machine Communication" tube indicates that such a change can be carried out by depressing the "Proceed with Routine" key.

When this key is depressed the following instruction appears:

```
YOU MAY AS FIRST ALTERNATE FOR ATLAS
CHANGE

  A.  LAST DELIVERY DATE
  B.  TOTAL NUMBER OF UNITS IN PROGRAM

OPERATOR:  USE ALPHA-NUMERIC KEYBOARD.
```

The staff decides not to change the number of units but to require that the last unit be delivered by January 1963 instead of the original June 1964. On the Alpha-Numeric Keyboard the letter "A" is punched, then the date January 1963.

On the Man-Machine Communication tube a statement appears to verify that this is indeed the change desired. In addition, on the left-hand tube under the heading of "First Alternate" the proposed last delivery date of January 1963 appears. The new instruction to the operator indicates that he can have the Atlas Program recomputed on the basis of this new delivery by depressing the "Proceed with Routine" key. However, if he made a mistake, he can "Cancel Keyboard Input" or for that matter he can "Cancel Last Instruction".

When the "Proceed with Routine" key is depressed the computer goes into a complex routine, based on the mathematical model developed for WSPACS.[1] Units to be delivered, squadrons projected and all expenditures for the Atlas program are re-computed on the basis of the proposed last delivery date of January 1963. In addition new totals for all Air Force programs are computed. This new information appears on the left-hand tube, tabulated under the old rows of information.

Now the staff has the choice of introducing this proposed change on the right-hand tube into the complete Air Force program, or make further Atlas trials. Results of various trials will appear simultaneously with the original plan on the left tube. After the staff has experimented with sufficient number of alternatives, they agree on a single proposed change for the Atlas program. This change is introduced on the right-hand tube into the Weapons Systems Program.

Now the staff is ready to proceed to another weapons system. Without going into the details of the actual exercise, we state that the go-ahead date and the last delivery date of certain programs can be changed. Some programs can be cancelled, or in others the numbers of units per squadron can be changed. In certain instances the phasing-out of weapon systems can be modified. In addition, it should be pointed out that certain subsidiary weapons systems programs are automatically changed as the major programs are changed. For instance, as the primary weapons systems programs are changed, the requirements for KC-135 changes, and these changes are introduced automatically into the system. In addition, capability is given to change the "Bomber to Tanker" ratio and "Bomber to GAM" ratio. It is seen then that as the analysis progresses the various weapons systems listed on the right tube are scanned and proposed changes are introduced. Through a step-by-step process a new weapons systems program is developed that is within fiscal limitations and meets the requirements imposed by the accelerated need for missiles.

In an actual demonstration on November 29, 1960, the following changes were made:

A. Accelerated Atlas Program by advancing the last delivery date from January 1964 to January 1963.

B. Moved the go-ahead date of the Minuteman from July 1961 to December 1960.

C. Cancelled the B-58.

D. Speeded up the phase out of the B-47 by reducing to 70 squadrons in fiscal year 1961, instead of 79 squadrons as originally planned.

As these changes were introduced, computations were carried out to show increases in expenditures for the Atlas and Minuteman Programs. Savings due to the cancellation of the B-58 were also computed. In addition, due to the cancellation of the B-58, reduction occurred in the quantities of B-58's and GAM's required. This resulted in savings in the KC-135 and GAM areas. Finally, the accelerated phase-out of the B-47 resulted in additional savings.

As a result of these actions, fiscal 1961 expenditures were brought within the revised expenditure limitations and the missile programs were accelerated.

It is to be emphasized that the exercise described here was carried out on the basis of a highly simplified mathematical model. Currently an effort is underway to improve the mathematical model by making it more realistic and flexible. However, it is expected that through the man-machine management system described here, the Air Force will be provided by a new increased capability in solving the difficult task of re-programming complex weapons systems programs.

Implementation Considerations

As of today there is no management system in operation patterned along the lines discussed in this paper. However, all elements required for the establishment of such a system are in existence and we believe that within a few years we will indeed see systems of this type operational. We wish to finish our paper by a brief discussion of problems of implementation and the general outlook for on-line management type systems.

Let us focus our attention to three fields of effort required for implementing such systems: (1) design of management systems in general,

(2) development of mathematical models, and (3) equipment considerations:

As far as the design of quantitative management control systems is concerned, during the last few years significant advances have been made.[2, 3] In particular, we refer to efforts like the Navy's "Program Evaluation Review Technique" (PERT) and the Air Force's "Program Evaluation Procedure" (PEP). Such advanced management techniques have shown significant success and there is today a substantial effort applied to extend these techniques. We recognize as one of the most significant weaknesses of current systems, that resource allocations and in particular financial considerations are not adequately treated yet. However, the critical need for such management systems exists and it is certain that significant further progress will be made within the next few years. Consequently, we believe that system design requirements for on-line type management systems could be met within a time span of about one to two years.

The second field of endeavor we want to talk about is the development of mathematical models. The system design work cannot be carried out without the appropriate mathematics. In the field of mathematical models significant progress is being made today[4, 5, 6] and it can be predicted with reasonable certainty that further progress will be made within the next few years. The type of mathematical model required for on-line management systems, has been only outlined in this paper and a great many of the details have not been worked out yet. In particular, the sub-optimization techniques required for the gaming exercise need further development. However, we believe that with a relatively small effort and short time, these mathematical models could be developed.

As far as equipment is concerned, we already stated that there is no system operating that could carry out all the required routines and input-output procedures. However, all the components are available and we see no significant difficulty in integrating existing components into a workable hardware system. More serious problems with respect to equipment are cost considerations. The financial benefits that can be obtained from on-line management systems is difficult to estimate and as a consequence it is difficult to determine how much money could be spent on equipment to create such management systems. However, aside from financial considerations, we believe that the equipment required could be manufactured within a one to two year time period.

It seems then that from the scientific and technological point of view, a management system of the type described in this paper could be created within a time period of one to two years. However, there is one further element to be considered. Traditional techniques of management control do not involve such sophisticated quantitative techniques as described in this paper. As a consequence of this, the design and implementation of advanced management systems must be accompanied by a parallel development in management philosophy. During the last few years there has been a significant shift in managerial concepts towards more sophisticated quantitative outlook. It is difficult to make a prognosis as far as management philosophy is concerned but it is difficult to believe that it will take more than two to three years to reach the appropriate management environment.

In summary, then, we estimate that it will be between two to five years before on-line management systems of the type described in this paper will become operational.

References

1. Saul Hoch, "Weapon System Programming and Control System, Mod 0 - A Demonstration Model," Operations Analysis Office, Directorate of Plans and Programs, Headquarters Air Material Command. Dec. 1960.

2. D. G. Malcom, J. H. Roseboom, C. E. Clark, W. Fazar. "Application of a Technique for Research and Development Program Evaluation," Operations Research, vol. 7 (1959), pp. 646-669.

3. "PERT/PEP Management Tool Use Grows," Aviation Week, (Nov. 28, 1960), pp. 85-91.

4. D. R. Fulkerson. "Increasing the Capacity of a Network: The Parametric Budget Problem," Management Science, vol. 5 (1959), pp. 472-483.

5. --- "A Network Flow Computation for Project Cost Curves," Management Science, vol. 7 (1961), pp. 167-178.

6. J. E. Kelley, Jr. and M. R. Walker. "Critical Path Planning and Scheduling," Proc. of the Eastern Joint Computer Conference, (1959), pp.160-173.

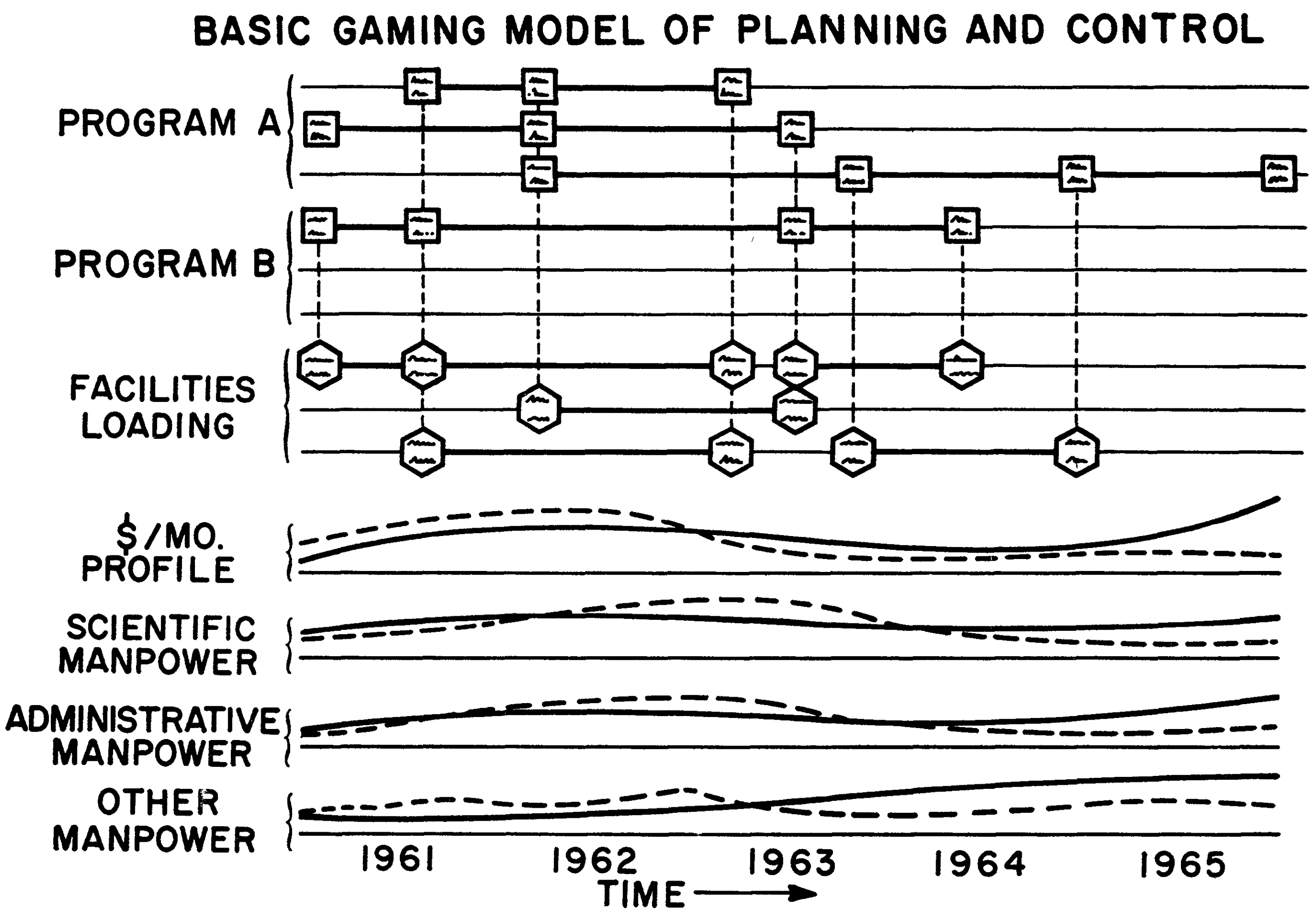


Figure 1. Basic Gaming Model

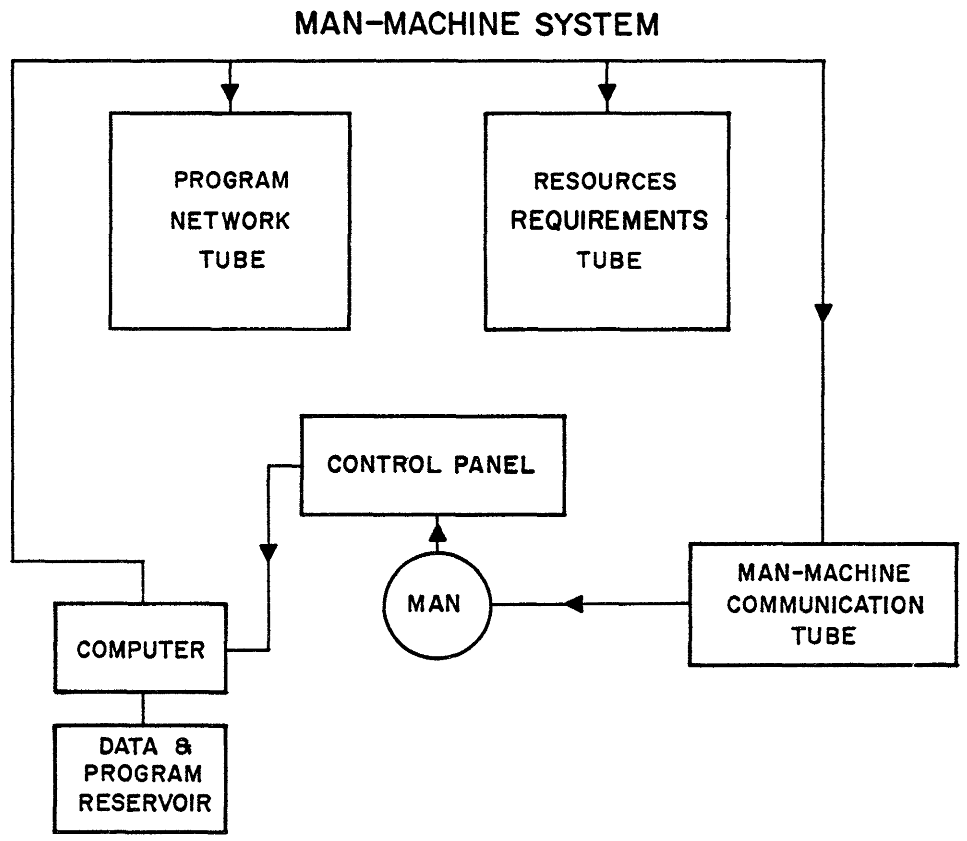


Figure 2. Man-Machine System

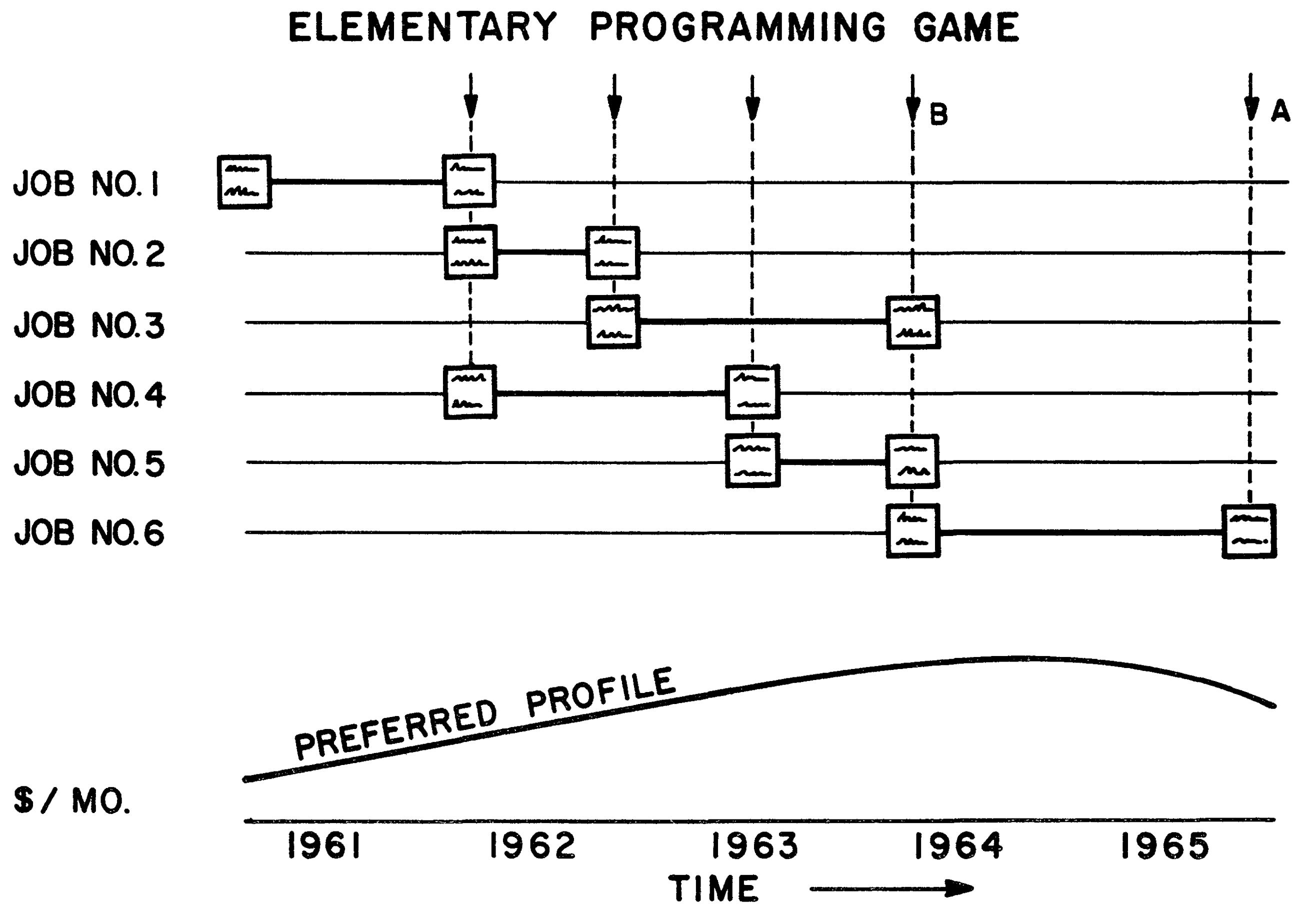


Figure 3. Elementary Programming Game

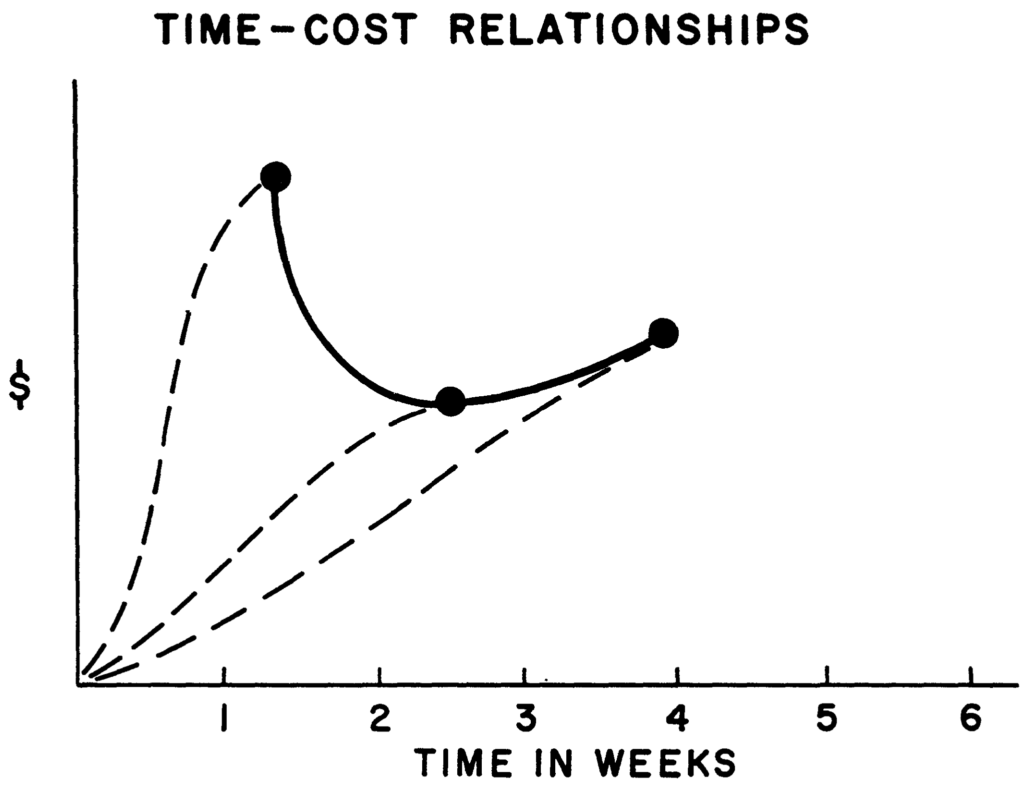


Figure 4. Time-Cost Relationships

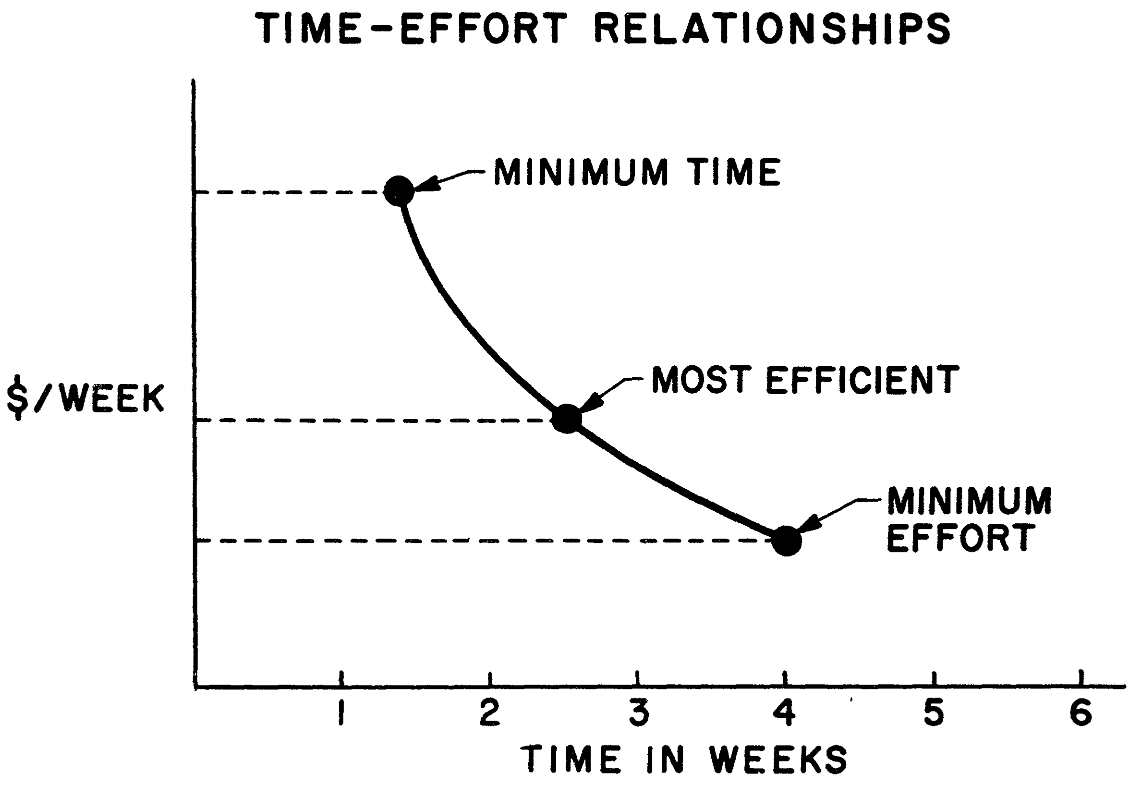


Figure 5. Time-Effort Relationships

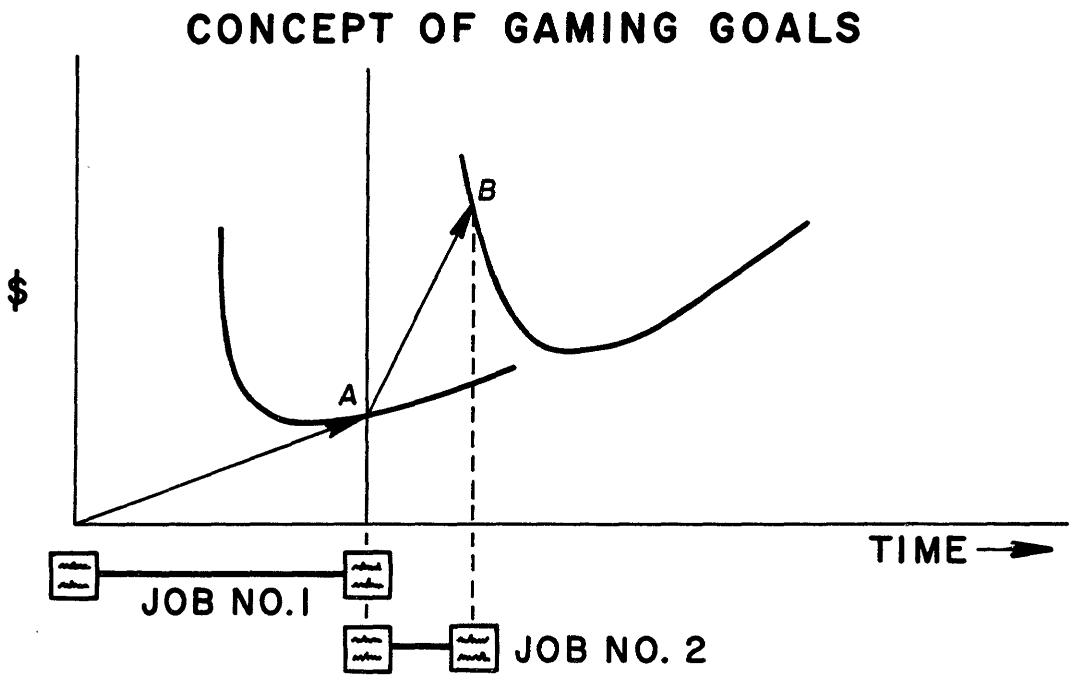


Figure 6. Concept of Gaming Goals

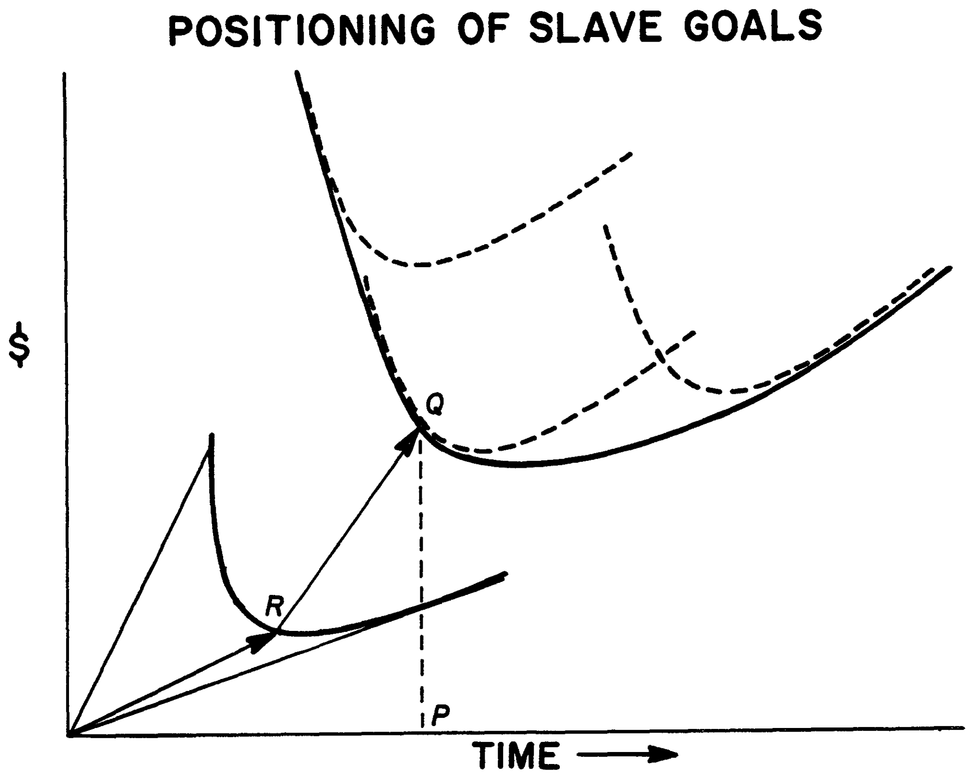


Figure 7. Positioning of Slave Goals

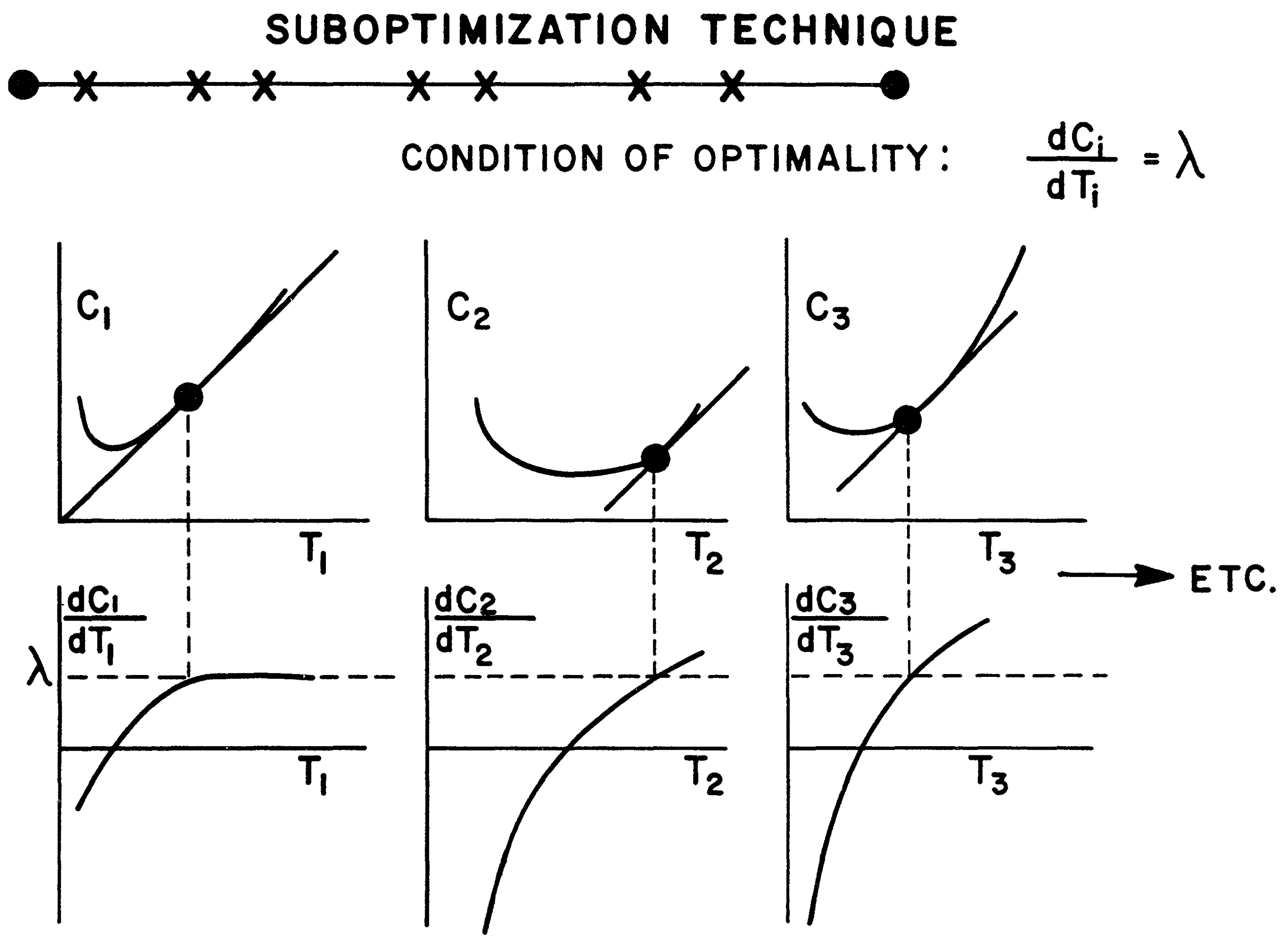


Figure 8. Suboptimization Technique

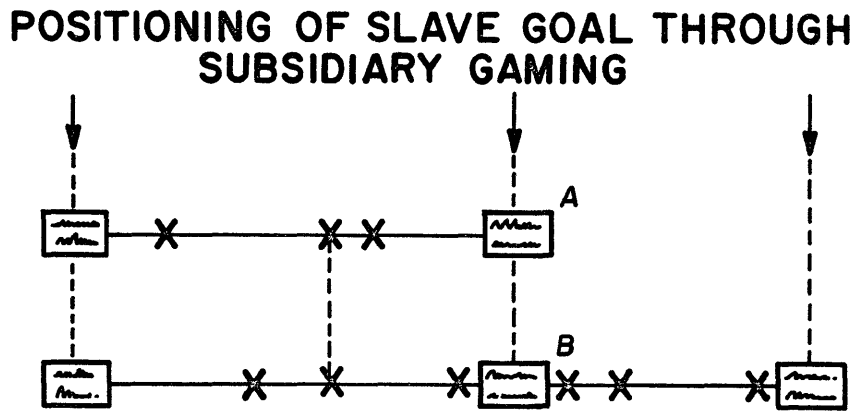


Figure 9. Subsidiary Gaming

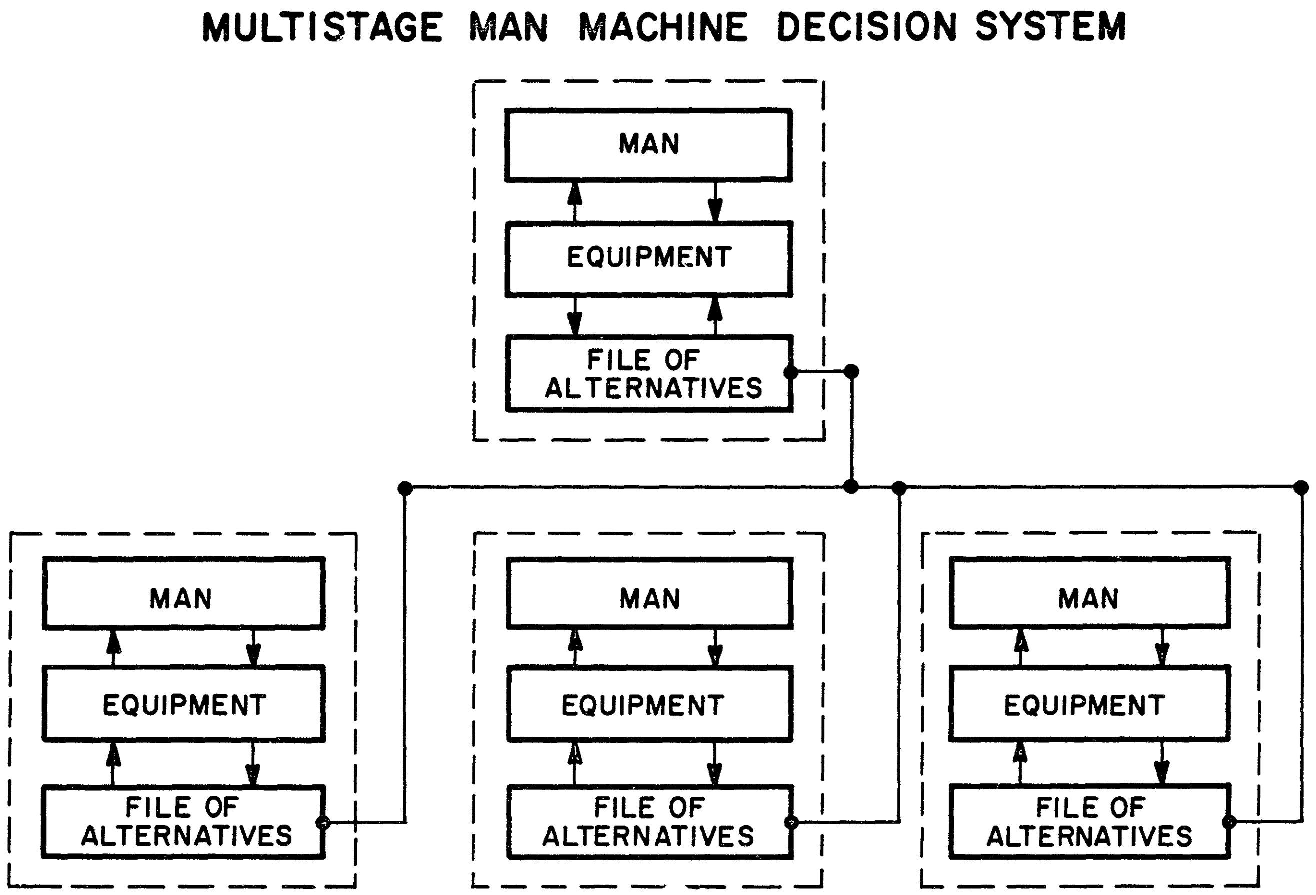


Figure 10. Multistage Gaming System

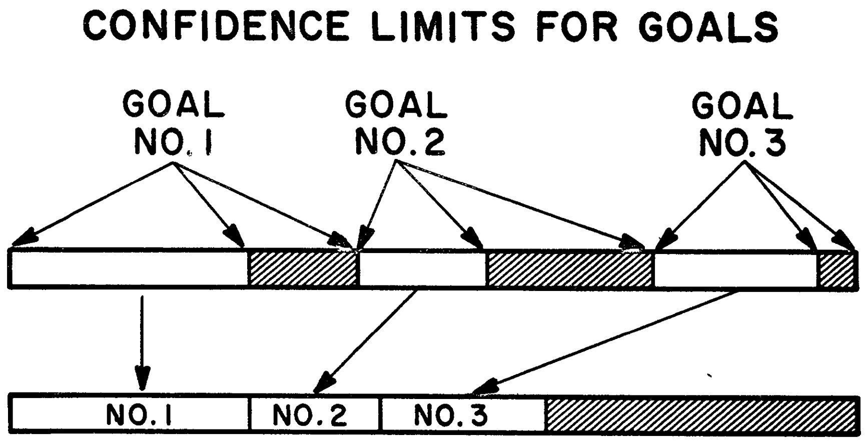
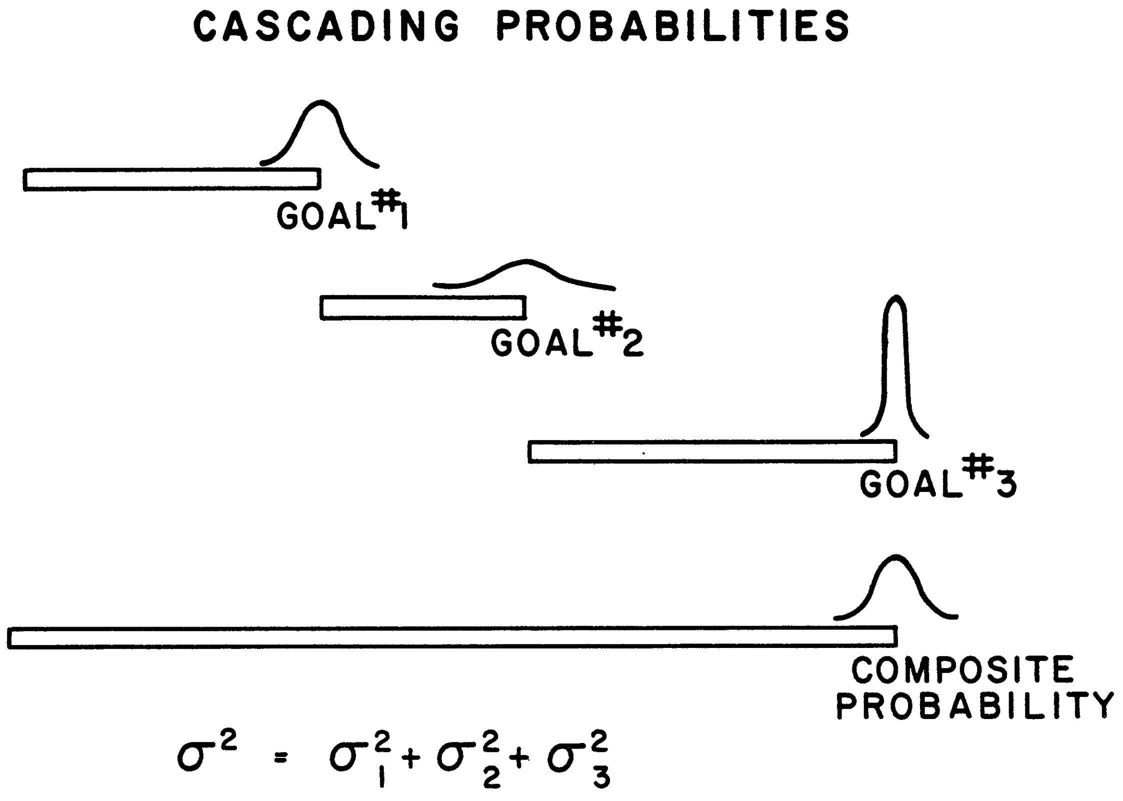


Figure 11. Confidence Limits



$$\sigma^2 = \sigma_1^2 + \sigma_2^2 + \sigma_3^2$$

Figure 12. Cascading Probabilities

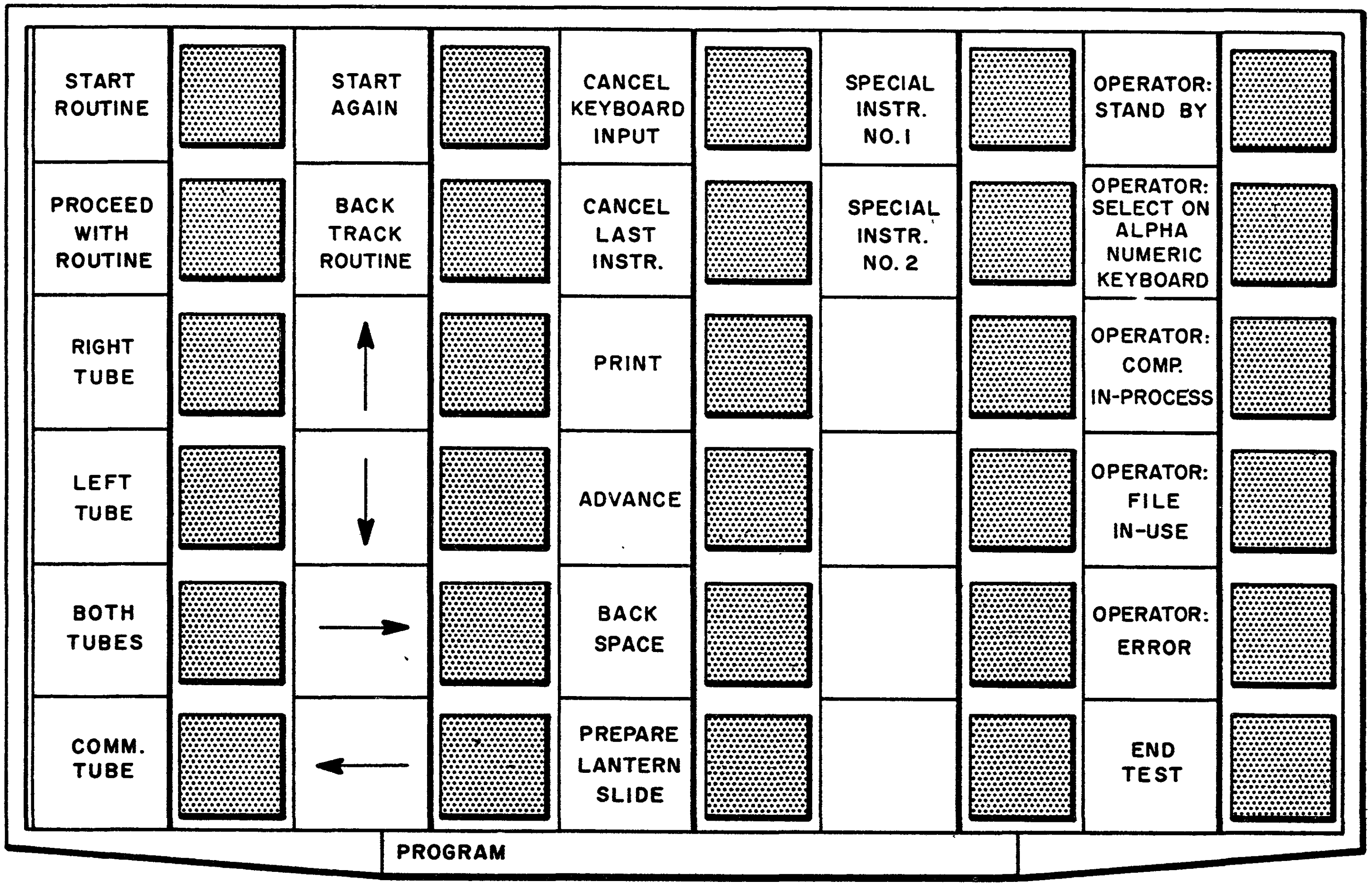


Figure 13. Control Panel

EXPENDITURES (Million Dollars)

		1961	1962	1963	1964	1965	Total 61-70
1. ATLAS	D and P	65	75	58	16		214
	Sys-conn	316	398	367	258	183	2437
2. MINUTEMAN	D and P		81	46	74	223	1245
	Sys-conn			10	39	102	1335
3. B-70	D and P			828	620	453	3115
	Sys-Conn					4	4013
4. B-52	D and P	95	25				120
	Sys-conn	1060	1149	1134	1134	1134	9931
5. B-58	D and P	58	25				83
	Sys-conn	159	324	309	309	309	2748
6. B-47	D and P						
	Sys-conn	915	741	510	197		2363
7. KC-135	D and P	163	87				250
	Sys-conn	174	230	230	230	230	2084
8. GAM-87	D and P			64	26	33	505
	Sys-conn					3	446
NON-SYSTEM		1200	1200	1200	1100	1100	
TOTAL FOR ALL WEAPONS SYSTEMS		4591	4795	5173	4367	4002	
LIMITATIONS		4600	4500				

Figure 14. Weapon Systems Expenditures

Total in Program	276	Total Delivery to Date	8	
Units per Squadron	12	Active Squadrons	0	
Go Ahead Date	May 1958	1st Alternate	2nd Alternate	3rd Alternate
Last Delivery	June 1964			

			EXPENDITURES				
			ATLAS		All Systems		
Year	To Be Delivered	Squadrons Projected	D&P	Sy. Con.	Non-Sys.	Total	Limitation
1961	49	4	65	316	1200	4591	4400
1st Alternate							
2nd							
3rd							
1962	138	11	75	398	1200	4795	
1st Alternate							
2nd							
3rd							
1963	230	19	58	367	1200	5173	
1st Alternate							
2nd							
3rd							

Figure 15. Atlas Program

APPLICATION OF DIGITAL SIMULATION TECHNIQUES TO HIGHWAY DESIGN PROBLEMS

Aaron Glickstein
Midwest Research Institute
Kansas City, Missouri

S. L. Levy
Midwest Research Institute
Kansas City, Missouri

Summary

A study of the operating characteristics of the driver-vehicle combination has yielded a general digital simulation model. This simulation model, which can duplicate traffic flow on a 17,000-ft. section of a freeway including two on-ramps and two off-ramps, can be used to economically evaluate alternate design criteria.

The simulated vehicle in the model, following decision rules based on actual traffic behavior, is allowed to maneuver through the section of freeway under study. The effects of changes in traffic volume, traffic velocity, freeway configuration, etc., can then be evaluated by noting changes in the computer output of traverse time, waiting time on ramp, volume-velocity relationship, weaving complexity, etc.

The computer simulation thus creates a duplication of the real situation at a small fraction of the cost of studying the real system. Furthermore, the simulation allows: (1) the evaluation of various freeway configurations without the expense of their construction, and (2) the performance of controlled experiments impossible to perform with the actual traffic.

Introduction

In recent years, because of the growth of the highway building program, traffic engineers have become concerned over the lack of knowledge about the nature of traffic flow. This lack of knowledge has hampered the design engineer in his ability to design an expressway which will move traffic smoothly and efficiently at a minimum cost. All too frequently after a highway has been opened it has been found that traffic problems arise which require extensive redesign and construction.

For example the following questions have still to be answered.

1. What should be the length of an acceleration lane?
2. What is the effect on traffic flow of locating interchange areas at various distances from one another.
3. What is the effect of various speed minimums and maximums on the traffic flow?

Analytical Models

During the past decade a number of descriptive theories concerning vehicular traffic have been put forward. In an attempt to classify these theories Haight[7] proposed three types.

The first, an analytical and deterministic model considers the geometrical and physical characteristics of the automobile and the assumed behavior of the driver. When specific values of velocity, acceleration, and vehicle following behavior are postulated, it is possible to describe the motion of a number of cars in one lane. The results, although precise, are limited to a small number of vehicles. Work in this area has been carried out by Pipes;[15] Chandler, Herman, and Montroll;[1] and Herman, Montroll, Potts, and Rothery.[9]

A second approach has been to consider traffic as a stochastic process and to treat it by the theory of queues. This approach has been utilized by Newell,[13] Tanner.[16] The theory of queues can, however, only be applied when vehicles move at essentially the same speed and when all vehicles enter the system at one point. Reasonable results have been obtained when traffic is actually queued, velocity uniform and the driver has few free decisions.

The third approach describes traffic flow in a continuum. This approach has been utilized by Newell,[11,12] Tse Sun Chow,[13] Greenberg,[6] and Lighthill and Whithan.[10] All three approaches are limited by the fact that it is assumed that vehicles do not overtake or pass each other.

Simulation Models

It is, however, possible to construct a simulation model for the study of traffic flow which is much more general than the three types mentioned. Simulation has been defined by Harling[8] as "the technique of setting up a stochastic model of a real system which neither oversimplifies the system to the point where the model becomes trivial nor incorporates so many features of the real system that the model becomes untractable or prohibitively clumsy."

Early work in the field of traffic simulation was carried out by Gerlough,[2] Goode,[4] Wong,[19] Trautman, Davis, et al.[17] Gerlough in a pilot study simulated two lanes of a highway system one-fourth of a mile long. In this simulation model the vehicles were allowed to choose their speeds from a three increment distribution of desired speeds. This program run on the SWAC computer required approximately 35 to 38 seconds of computer time to simulate one second of real time.

Goode[5] in his simulation model represented a single intersection of a simple type. Provisions were made for the origination of cars in lanes approaching the intersection, and for a stochastic mechanism for the determination of the direction of travel of any particular car entering the intersection (i.e., right, straight ahead, or left). The measure of effectiveness used was the average delay per car. The ratio of computer time to real time was 3:2.

Although these early models were of rather limited extent, they indicated that simulation procedures could be used to run controlled experiments in order to gain a better understanding of the nature of traffic flow.

The Expressway Interchange Model

In 1958 the Midwest Research Institute, under a grant from the Bureau of Public Roads, started a research program on the simulation of expressway traffic flow. The first phase of this program was devoted to the simulation of an on-ramp area of the expressway.[14] That model simulated a 1,700-ft. section of a three lane expressway including one on-ramp. Based on the experience of this pilot study a more general model was developed. That simulation model is described in this paper.

The Study Area

The portion of the expressway under study is set up in a 4 x N matrix ($N \leq 999$), Fig. 1. The four rows represent: (1) the three through lanes 1, 2, and 3, and (2) the ramp, acceleration, and deceleration Lane 5. Each of the N blocks represents a 17-ft. section of freeway, the approximate length of an automobile. For the simulation runs on the computer, any value of ($N \leq 999$) can be utilized. Locations of interchanges are designated as follows:

A = ramp input location
B = nose of on-ramp
C - end of acceleration lane
D = beginning of deceleration lane
E = nose of off-ramp
F = off-ramp output location

The program is very flexible and permits the on- and off-ramps to be located at any point on the section of freeway under investigation.

Input Factors for Simulation Model

In order to obtain a true duplication of actual traffic behavior on the freeway the simulation model should contain all factors which influence traffic behavior. In this model the following factors are considered:

1. The volume of entering and exiting vehicles.
2. The distribution of vehicles to lanes.
3. The velocity distribution of vehicles.
4. The gap acceptance distribution of merging and weaving vehicles.
5. The acceleration of entering vehicles.
6. The deceleration of exiting vehicles.
7. The distribution to lanes of exiting vehicles.

In addition, all vehicles are allowed to shift lanes in order to pass slower moving vehicles in front of them.

Simulation Procedure

Basically, the procedure consists of simulating the arrivals of cars into the section of highway under consideration and then controlling the action of the vehicle by a set of decision processes. During each second of real time each vehicle in the matrix is examined. The vehicle is allowed to advance, weave, merge, accelerate, decelerate, or exit according to logical rules describing the behavior of actual vehicle-driver combinations. Just prior to examining all vehicles at each second, each of the input locations is evaluated. Inspection starts at vehicles closest to the end of the section of highway under examination and proceeds to vehicles in the input location.

A description of the over-all logic involved in processing a vehicle through the system is given in the flow diagram (Fig. 2). Detailed flow diagrams for the computer can be obtained in Reference No. 3.

The following parameters are used in the flow diagrams:

V = total volume in Lanes 1, 2, and 3; vehicles per hour

V_i = vehicles per hour in the ith lane

$$\sum_{i=1}^{3} V_i = V$$

B_n = block number of vehicle under inspection

${}_iB_{n-1}$ = block number of vehicle in ith lane preceding vehicle under inspection

${}_iB_n$ = block number of vehicle in ith lane parallel to or behind vehicle under inspection

v = velocity of vehicle under inspection

${}_iv_n$ = velocity of vehicle in ith lane, parallel to or behind vehicle under inspection

W = time gap

Simulation Output

The present model was programed so that the following information can be obtained about each simulation run:

1. The volume of vehicles traversing the system in each lane.
2. The volume of vehicles entering the freeway through each on-ramp.
3. The volume of vehicles exiting the system at each off-ramp.
4. The number of vehicles which stop on the acceleration lane.
5. The length of the queue on the acceleration lane.
6. The number of vehicles that desire to exit but cannot.
7. The distribution of through-vehicle traverse times.
8. The distribution of ramp vehicle traverse times.
9. The average vehicle velocity in each lane.
10. The number of weaves from:
    a. Lane 1 to 2
    b. Lane 2 to 1
    c. Lane 2 to 3
    d. Lane 3 to 2

Input Data Preparation

Computer Operating Note

To operate the program, an IBM 704 computer is needed which has at least 8,192 words of core storage. One magnetic tape unit is needed (Logical Tape 0), that one being used to produce an output tape for printing on an off-line printer.

On the Computer Console, Sense Switch 4 should be depressed if it is desired to obtain in the problem output a listing of all problem input parameters.

To run the program, assemble the program deck with a series of Control Cards immediately following the deck. The number and type of Control Cards will control the analyses to be run.

There are 13 different Control Cards that can be entered, which insert the necessary setup data describing the system configuration and velocity and weaving distributions. Appropriate Control Cards should immediately follow the program deck. A Problem Card, containing data peculiar to each analysis then follows and, after this Problem Card, change cards for any of the control cards and other Problem Cards may be entered to control the running of a series of traffic problems.

Control Card Format

The 13 Control Cards are divided into four groups. Five cards provide the velocity distribution data for Lanes 1, 2, 3, and the two possible on-ramps. A second group of five cards provide data concerning gap acceptance for weaving operations. A single card specifies the geometry of the model; the length of the through lanes, the ramp entrance and acceleration lane geometry for both on-ramps, and the deceleration lane geometry and ramp exit for both off-ramps.

A fourth group, containing two cards give data concerning the exiting decision process; one card for off-ramp No. 1 and the other card for off-ramp No. 2.

The Problem Card contains a test of identification number, the length of time the analysis is to be run, the volume of through traffic and the input volume to each of the two on-ramps.

Detailed description of Control Card format is given in Reference No. 3.

Study of Interchange Configuration

Various types of controlled experiments can be carried out using this simulation model. For example, experiments on the effects on traffic flow of (1) various on-ramp vehicle volumes, (2) various acceleration lane lengths, (3) various velocity distributions, (4) various geometric configurations, and (5) combinations of the above, can be carried out with this model.

Experiments have been carried out on the effect on traffic flow of spacing between an on-ramp and an off-ramp, under varying traffic volumes.

Interchange Configuration

Two interchange configurations (Fig. 3) were examined. Each configuration was 200 blocks or 3,400 ft. long and contained one on- and off-ramp combination. Each acceleration and deceleration lane was 595 ft. long. In Configuration I, exiting vehicles have 2,465 ft. to travel to the nose of the off-ramp while in Configuration II the distance is 3,230 ft. In Configuration I, the distance between the acceleration and deceleration lane is 340 ft. and 1,870 ft. in Configuration II. All exiting vehicles were designated at block number 199.

Input Data

Volume of traffic: For both configurations, the input to the simulation was for 750 ramp vehicles per hour. For Configuration I experiments with through-lane volumes of 2,000, 3,000, 4,000, 5,000 and 6,000 vehicles per hour were conducted. For Configuration II experiments with through-lane volumes of 1,000, 2,000, 3,000, 4,000, 5,000 and 6,000 were conducted. Two tests were conducted at each volume.

Distribution of volume to lanes: In all experiments carried out, the traffic volumes were assigned to the three lanes according to the following relationships:

$$P_1 = 0.43693 - 0.22183\alpha + 0.05730\alpha^2 - 0.00046\alpha^3 \qquad (1)$$

$$P_2 = 0.48820 - 0.03136\alpha + 0.00006\alpha^2 + 0.00024\alpha^3 \qquad (2)$$

$$P_3 = 0.07487 + 0.25319\alpha - 0.05736\alpha^2 + 0.00382\alpha^3 \qquad (3)$$

where

P_i = proportion of total volume in the ith lane

α = total freeway volume in thousands of vehicles per hour

Velocity distributions: Three different velocity distributions were utilized in the simulations. One velocity distribution was used for the on-ramp vehicles, a second for the vehicles in Lane 1 and a third for vehicles in Lane 2 and Lane 3. These velocity distributions are presented in Table I.

TABLE I

Input Velocity Distributions

Vehicle Velocity (blocks/sec.)	Cumulative Per Cent		
	Ramp Lane	Lane 1	Lanes 2 and 3
1.50	0.026	0.001	0.003
2.00	0.061	0.010	0.015
2.38	0.116	0.040	0.034
2.81	0.314	0.180	0.064
3.25	0.683	0.435	0.138
3.69	0.888	0.737	0.322
4.13	0.979	0.899	0.759
4.56	0.994	0.989	0.970
5.00	0.998	0.999	0.998
5.44	1.000	1.000	1.000

Gap acceptance distributions: Three different gap acceptance distributions were utilized in the simulation tests. The distributions for merging vehicles (both stopped and moving) are presented in Table II. The gap acceptance data for vehicles weaving between lanes are presented in Table III.

TABLE II

Gap Acceptance for Merging Vehicles

Size of Gap (sec.)	% of Total No. of Moving Vehicles Accepting Gap	% of Total No. of Stopped Vehicles Accepting Gap
0.00-1.00	12.5	0.0
1.01-2.00	56.8	2.8
2.01-3.00	77.0	17.4
3.01-4.00	95.1	36.0
4.01-5.00	97.0	64.9
5.01-6.00	98.0	95.0
6.01-7.00	100.0	100.0
7.01-8.00	100.0	100.0
8.01-9.00	100.0	100.0
9.01-10.00	100.0	100.0

TABLE III

Gap Acceptance Distribution for Weaving Vehicles

Length of Gap (w) (sec.)	Cumulative Probability of Acceptance
0.00-0.25	0.00
0.26-0.50	0.00
0.51-0.75	0.00
0.76-1.00	0.00
1.01-1.25	0.10
1.26-1.50	0.30
1.51-1.75	0.60
1.76-2.00	1.00

Exiting vehicles: For both freeway configurations 10 per cent of the through vehicles were designated as exiting vehicles. These exiting vehicles were further allocated to the three lanes according to the following schedule:

1. 90 per cent of exiting vehicles in Lane 1 at Block No. 199;
2. 9 per cent of exiting vehicles in Lane 2 at Block No. 199; and
3. 1 per cent of exiting vehicles in Lane 3 at Block No. 199.

Experimental Results

The controlled experiments described in the previous section were carried out on a 704 digital computer. The ratio of computer time to real time varied from 1 to 5 for low traffic volumes, to almost 1 to 1 for the higher volumes.

The over-all results of the experiments indicated that there were no significant effects on the traffic flow patterns of the freeway as a result of this change in geometric configuration. A comparison between the through vehicle traverse times for Configurations I and II are shown in Fig. 4. The effect of increased volume on the number of weaves between lanes is shown in Fig. 5. The computer output also indicated that the number of vehicles stopping on the acceleration lane increased with increased volumes of traffic, Fig. 6.

Conclusions and Recommendations

This study has shown that digital simulation can be used to faithfully duplicate actual traffic flow in an on- off-ramp area of a freeway. The output of the simulation, furthermore, gives measures of effectiveness which can be used to evaluate alternate highway designs.

The simulation experiments performed indicate the need for research and experimentation in a wide variety of areas to answer questions such as the following:

1. What is the effect of various vehicle distributions to lanes on the traffic flow?
2. What is the effect on traffic flow of various distances between adjoining on-ramps?
3. What is the effect of various desired velocity distributions on traffic flow?
4. At what volume of traffic do weaving movements between lanes become hazardous?
5. What is the effect on traffic flow of various volumes of commercial vehicles?

The simulation model developed in this study can serve as an efficient tool to answer these and other problems in the continuous quest for means of moving traffic safely and efficiently.

Acknowledgement

The authors would like to acknowledge the valuable work contributed by Mr. L. Findley in programing the computer.

Bibliography

1. Chandler, Herman and Montroll, "Traffic Dynamics; Studies in Car Following," Operations Research Journal, March, April, 1958.

2. Gerlough, D. L., "Simulation of Freeway Traffic by an Electronic Computer," Proceedings of the Highway Research Board, Vol. 35, 1956.

3. Glickstein, A., Findley, L. D., and Levy, S. L., "A Study of the Application of Computer Simulation Techniques to Interchange Design Problems. Paper presented at 40th Meeting of Highway Research Board, January 9-13, 1960.

4. Goode, H. H., and True, W. C., "Vehicular Traffic Intersections," paper presented at the 13th National Meeting of the Associat tion for Computing Machinery, June 11-13, 1958.

5. Goode, H. H., "The Application of a High-Speed Computer to the Definition and Solution of the Vehicular Traffic Problem," Operations Research Journal, December, 1957.

6. Greenberg, H., "Analysis of Traffic Flow," Operations Research Journal, January, February, 1959.

7. Haight, F. A., "Towards a Unified Theory of Road Traffic," Operations Research Journal, November, December, 1958.

8. Harling, J., "Simulation Techniques in Operations Research," Operations Research Journal, May, June, 1958.

9. Herman, Montroll, Potts, and Rothery, "Traffic Dynamics: Analysis of Stability in Car Following," Operations Research Journal, January, February, 1959.

10. Lighthill and Whitham, "On Kinematic Weaves, I and II," Proceedings - Royal Society of London, May, 1955.

11. Newell, G. F., "Mathematical Models for Freely Flowing Highway Traffic," Brown University, December, 1954.

12. Newell, G. F., and Bick, J. H., "A Continuum Model for Traffic Flow on an Undivided Highway," Division of Applied Mathematics, Brown University, July, 1958.

13. Newell, G. F., "Statistical Analysis of the Flow of Highway Traffic Through a Signalized Intersection," Quarterly of Applied Mathematics, January, 1956.

14. Perchonok, P., and Levy, S. L., "Application of Digital Simulation to Freeway ON-RAMP Traffic Operations," Proceedings of the Highway Research Board, Vol. 39, 1960.

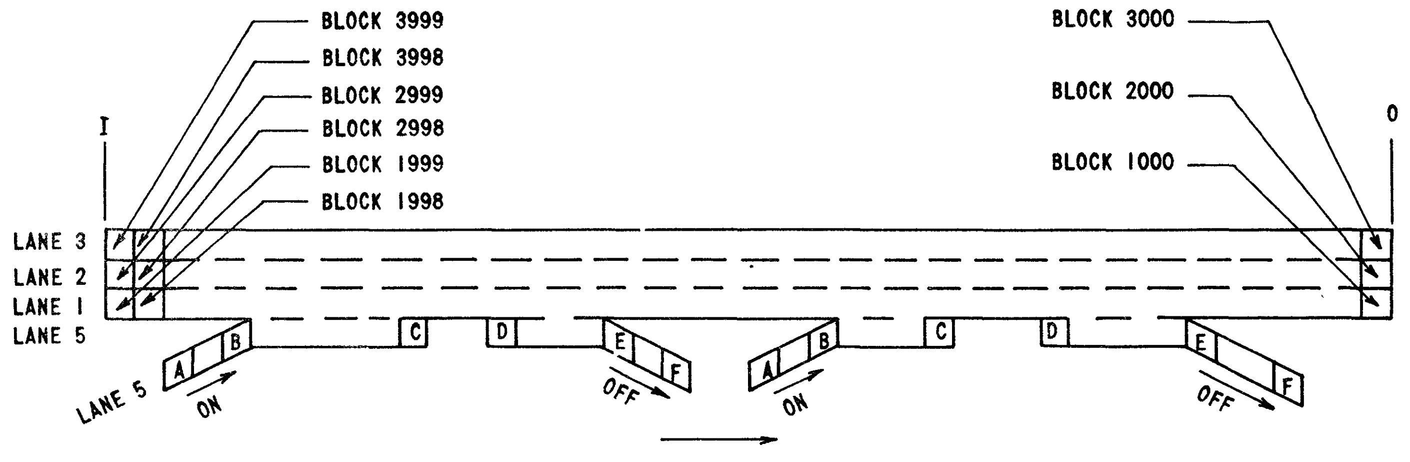
15. Pipes, L. A., "Operational Analysis of Traffic Dynamics," Journal of Applied Physics, March, 1953.

16. Tanner, J. C., "A Problem of Interference Between Two Queues," Biometrika, June, 1953.

17. Trautman, D. L., Davis, H., Heilfron, J., Er-Chun Ho, Mathewson, J. H., and Rosenbloom, A., "Analysis and Simulation of Vehicular Traffic Flow," Institute of Transportation and Traffic Engineering, University of California, Research Report No. 20, 1954.

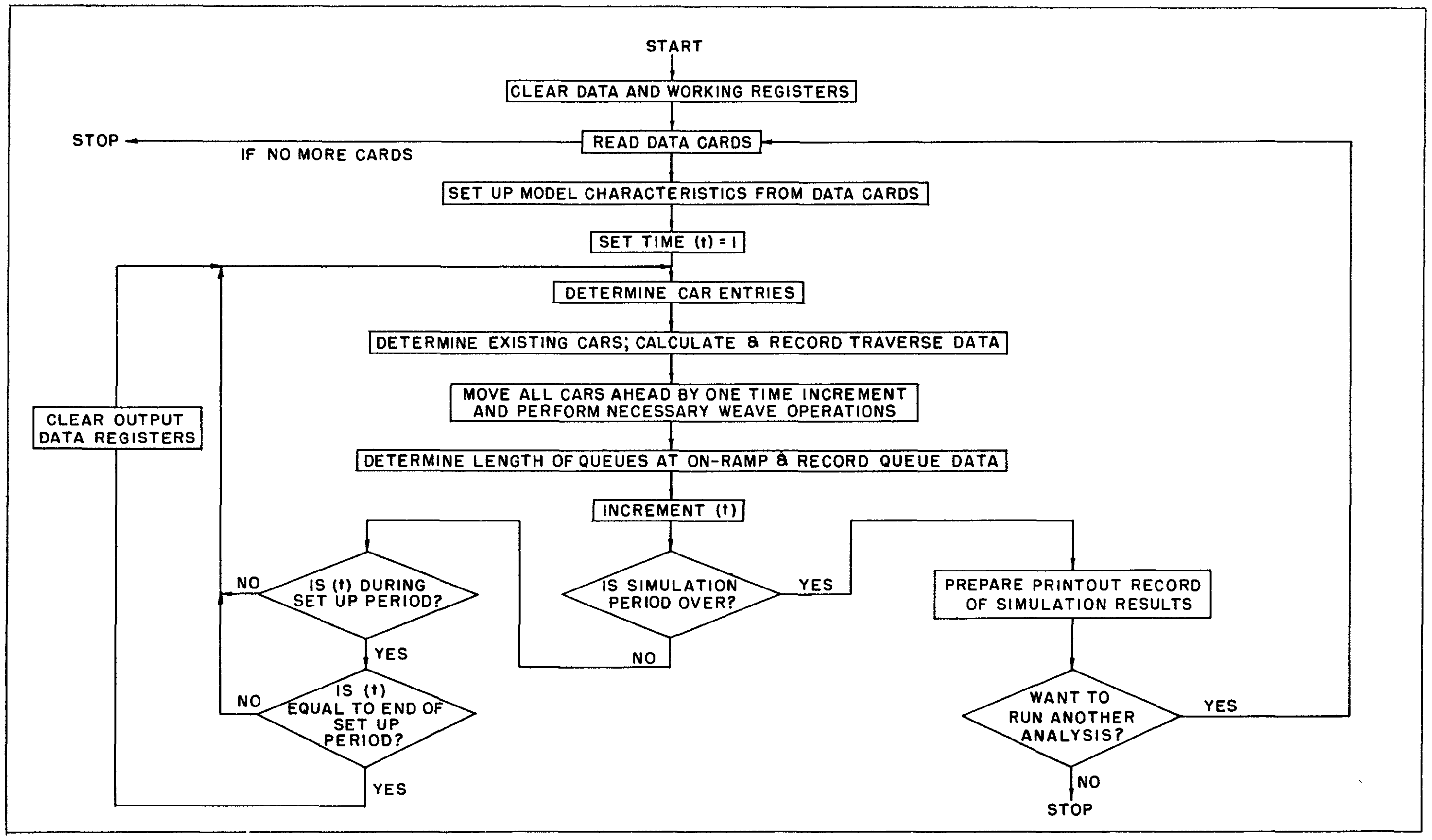
18. Tse-Sun Chow, "Operations Analysis of a Traffic Dynamics Problem," Operations Research Journal, November, December, 1958.

19. Wong, S. Y., "Traffic Simulation with a Digital Computer," Proceedings - Western Joint Computer Conference, February, 1956.



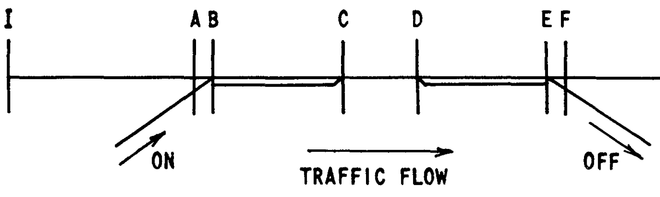
WHERE A - RAMP INPUT LOCATION
B - NOSE OF ON-RAMP
C - END OF ACCELERATION LANE
D - BEGINNING OF DECELERATION LANE
E - NOSE OF OFF-RAMP
F - OFF-RAMP OUTPUT LOCATION
I - THROUGH LANE INPUT
O - THROUGH LANE OUTPUT

Study Area Matrix for Computer Simulation of an Interchange Area


START
CLEAR DATA AND WORKING REGISTERS
READ DATA CARDS
STOP
IF NO MORE CARDS
SET UP MODEL CHARACTERISTICS FROM DATA CARDS
SET TIME (t) = 1
DETERMINE CAR ENTRIES
DETERMINE EXISTING CARS; CALCULATE & RECORD TRAVERSE DATA
MOVE ALL CARS AHEAD BY ONE TIME INCREMENT AND PERFORM NECESSARY WEAVE OPERATIONS
CLEAR OUTPUT DATA REGISTERS
DETERMINE LENGTH OF QUEUES AT ON-RAMP & RECORD QUEUE DATA
INCREMENT (t)
IS (t) DURING SET UP PERIOD?
NO
YES
IS (t) EQUAL TO END OF SET UP PERIOD?
NO
YES
IS SIMULATION PERIOD OVER?
YES
NO
PREPARE PRINTOUT RECORD OF SIMULATION RESULTS
WANT TO RUN ANOTHER ANALYSIS?
YES
NO
STOP

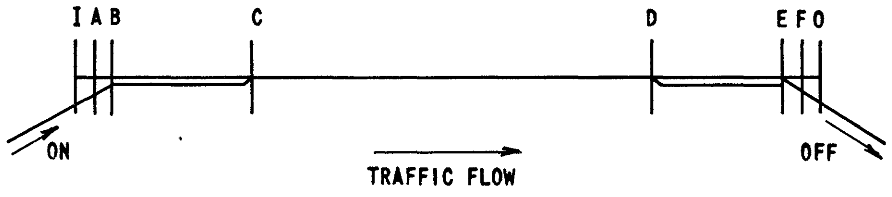
Flow Diagram of Over-all Computer Logic

CONFIGURATION I



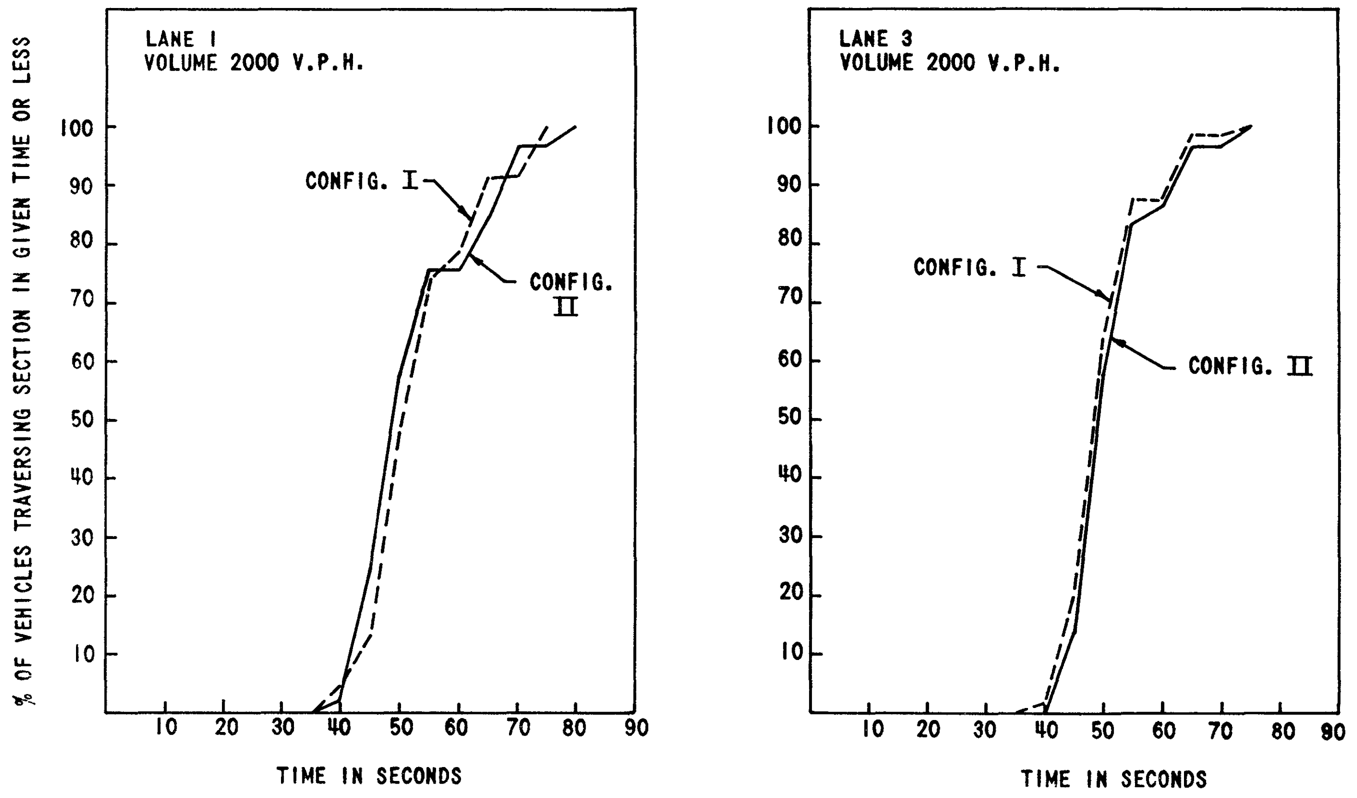
BLOCK	DISTANCE
I = 200	3400 FT
A = 150	2550 FT
B = 145	2465 FT
C = 110	1870 FT
D = 90	1530 FT
E = 55	935 FT
F = 50	850 FT
0 = 00	000 FT

CONFIGURATION II

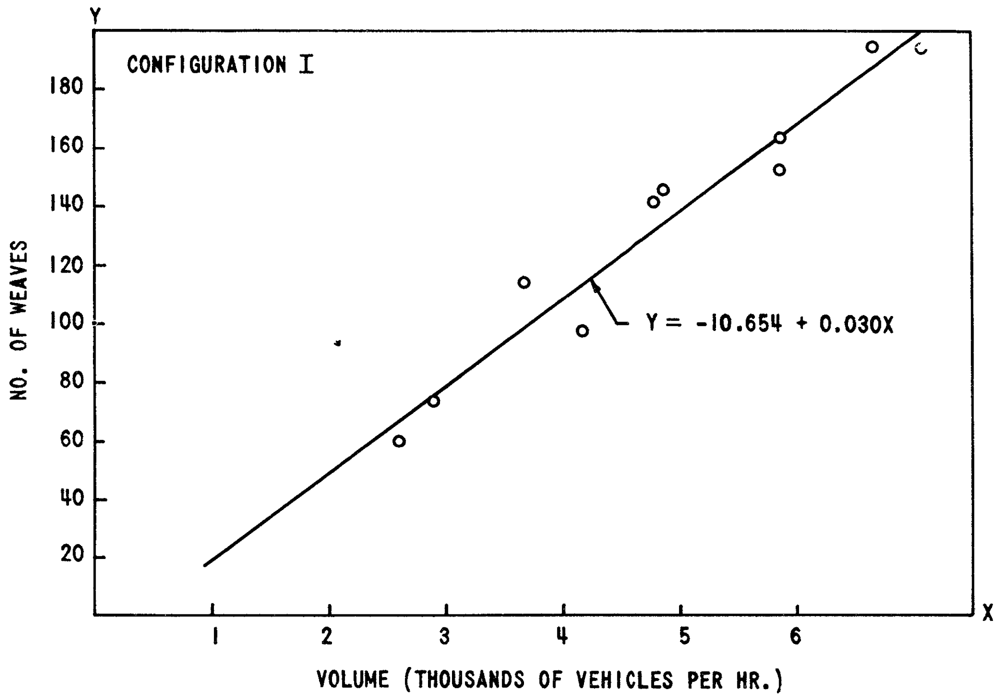


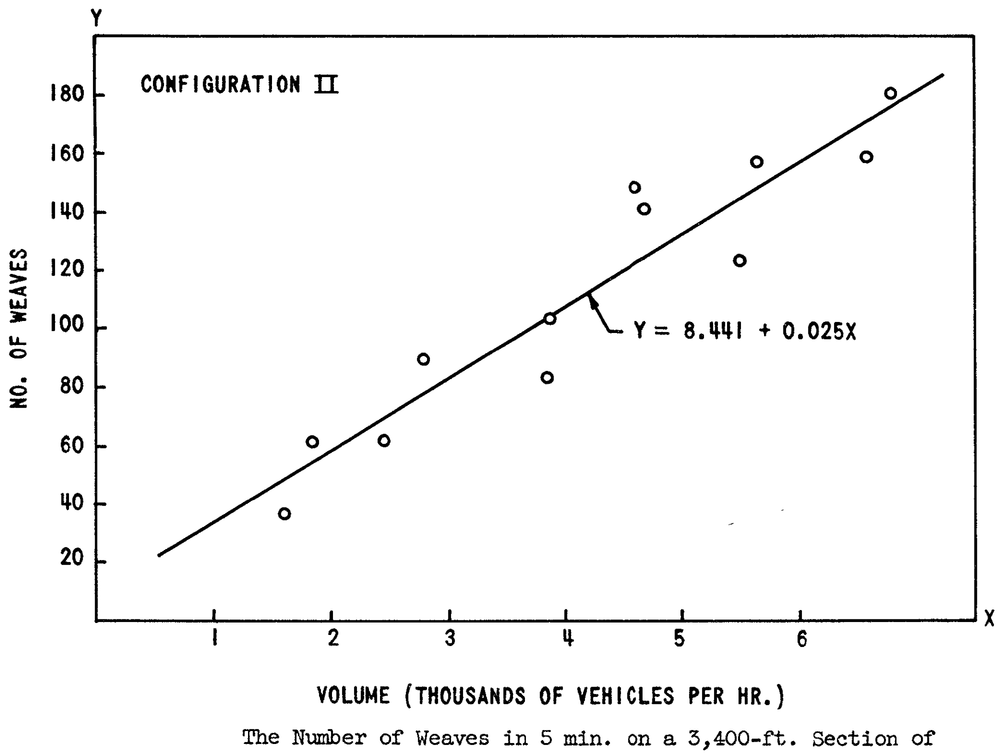
BLOCK	DISTANCE
I = 200	3400 FT
A = 195	3315 FT
B = 190	3230 FT
C = 155	2635 FT
D = 45	765 FT
E = 10	170 FT
F = 5	85 FT
0 = 00	00 FT

The Configuration of the Two Sections of Freeway Being Examined

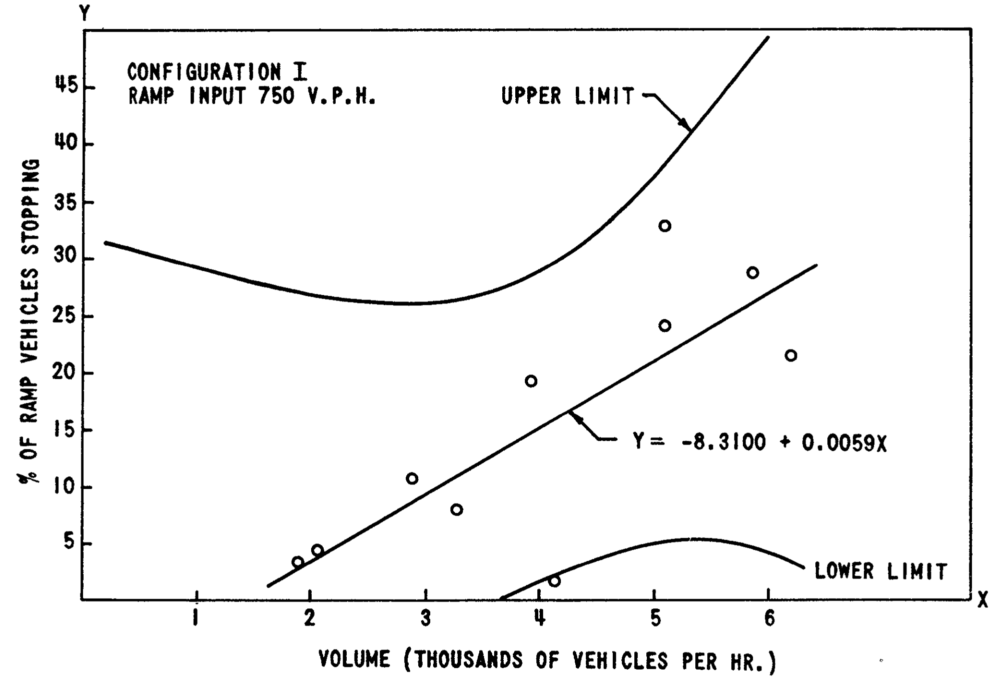


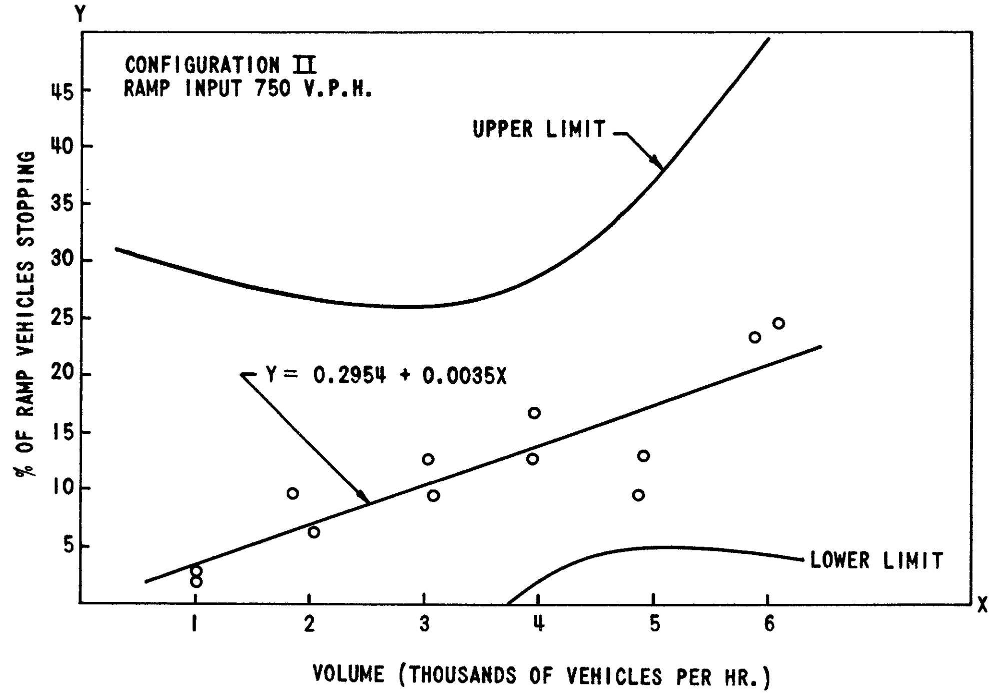
Cumulative Distribution of Through Vehicle Traverse Times at a Freeway Volume of 2,000 V.P.H.





The Number of Weaves in 5 min. on a 3,400-ft. Section of Freeway as a Function of Total Through Volume





The Relationship Between the Per Cent of Vehicles Stopping on Acceleration Lane (Y) and the Total Through Volume (X)

THE USE OF MANNED SIMULATION IN THE DESIGN OF AN OPERATIONAL CONTROL SYSTEM

M. A. Geisler
W. A. Steger

Logistics Department
The RAND Corporation
Santa Monica, Calif.

Summary

This paper describes the general features of the planning and operations phases of a new weapon system. The uncertainties inevitable in planning mean that considerable effort is made during the operations phase to adjust the weapon system and its resources to the actual environment it finds so as to attain the desired level of operational capability. The adjustment mechanism is called an operational control system in this paper. Elements of such an operational control system are described.

The proposal is made that a better control system can be designed if simulation is used to help design it during the planning phase. The use of simulation will not only produce a better control system earlier but it will permit the planners to adjust the other resources provided for the weapon system so that they are compatible with the environment and the control system. An example of such a study is described in this paper.

I. Introduction

The military structure is going through tremendous technological change. Successive generations of new weapon systems are coming along at a fast rate, and each contains such different characteristics that in many ways past experience is of little use. Furthermore, the competition among nations in military technology is causing efforts to reduce the time length of the defense planning cycle, which complicates the necessary process of coordination and interaction that is required to produce consistent and compatible hardware, operational plans, and support plans. These two factors, compression in planning and rapid technological change, mean that the planning process is becoming more difficult and its results therefore less dependable. The purpose of this paper is to suggest that the design of better operational control systems is one way of contending with this planning difficulty, and further, that the design of such systems will be considerably aided by the use of simulation techniques. In developing this thesis, this paper covers first a discussion of such a system, the use of simulation for designing a specific system, an example of such a designed system, how the simulation of the operational process was used to improve the planning decisions, the role of computers in the simulation and finally, some of the limitations of the technique.

II. Need for an Operational Control System

It takes a long time to develop, procure, and install modern weapons that are needed to satisfy a future operational requirement. In Chart 1, we try to define in general terms the process that leads to the creation of such systems. As can be seen, there are two phases: a planning phase and an operational phase. The planning for new weapon systems takes place over several years. It usually begins with a formulation of the future environment for which a new weapon is to be developed. This environment may describe the operational situation, or requirements foreseen, and possible enemy intentions and capability. This environment is then used as a guide to define the hardware characteristics of the projected weapon system, the operational plans for use of the weapon, and the logistics or support plan of the weapon for supporting its operation. This planning involves many organizations, many people, and as we have said, takes a long time, perhaps several years, because it is an involved process.

Necessarily, many of the factors developed for planning use are subject to much uncertainty, and it is very difficult to define all the relevant relationships affecting both the operating and support characteristics and functions of the weapon system. Nevertheless, these plans are used to determine resource and budget estimates for the new weapon system. These estimates then go through a budget and procurement cycle which takes additional years. We thus see that the planning for new weapon systems involves many organizations, and requires many assumptions about the uncertain future, technical capabilities, and enemy intentions.

The operations phase begins when the resources in the form of weapons, people, equipment, etc., resulting from the planning cycle are produced. These resources are now confronted with the actual environment. For many reasons, there are many difficulties involved in fitting the actual environment and existing resources together to achieve a satisfactory level of operational capability. For one thing, the predicted environment and the actual environment differ simply because of predictive errors. The enemy capabilities are now different from what we expected; the operational situation is either more or less demanding than that predicted; furthermore, the actual resources may differ appreciably both in quality and quantity from those planned. The budget cycle provided funds for less resources than expected; the reliability of the weapon system is not as great as projected; and the cost of the weapons and other resources are greater than planned.

Thus, the major activity during the operations phase is to mesh and adjust the actual environment and the existing resources so that the resulting operational capability is as great as possible. The system performing this meshing and adjusting is called an operational control system. It performs this activity, first, by using the resources available at any one time to obtain as high a level of operational capability as possible, and second, by specifying the adjustments in future resources that will improve operational capability. Thus, the operational control system enters directly into the determination of both operational capability and efficient resource determination and utilization. In this respect, the control system can be treated as another resource of the total system; and therefore, its contribution to operational capability and its cost can be weighed against that of all other resources.

With this in mind, there are three main points we should like to make about an operational control system. First, it is an integral part of the process that determines operational capability, and its use is an important means for contending with planning uncertainty. Second, it is a complex system in its own right, and its creation may take a long time and much resources. Third, if one has a means of predicting the operational capability, this information can be used during the planning cycle to adjust resource requirements. A more responsive, accurate, and comprehensive control system may require less total resources to achieve a given level of operational capability.

It is the premise of this paper that simulation early in the life of a weapon system can predict (to some degree) the contribution of the operational control system to the preferred mix of resources (including the control system) for the operational weapon. If the composition of this preferred mix can be known during the planning process, such information can influence resource policies and future requirements.

III. General Description of an Operational Control System

The control systems that are being developed or conceived by the Air Force to enable major commands to control lower echelons have characteristics which far transcend the hardware elements that such systems must necessarily possess. They are man-machine systems, and although the requirements and availability of hardware to be used in such systems is obviously very relevant, the demands upon the man in the system are at least as important to determine.

Chart 2 helps to describe a simplified version of the elements and functions of such an operational control system. At any given point in time, the status of the environment and the existing resources represent what we call the current system status. These include such data as the operational condition of the weapon, work assignments of personnel, equipment location and uses, etc. The operational control system works on this status to improve the future capability of the weapons system. Since the actual status may be a complex of vast geographic distances, thousands of personnel, many million dollars of weapons, equipment, and facilities, the control system creates an abstract representation to help it to manipulate and to improve the actual system.

This abstract representation is obtained through an information system which receives inputs as changes in the current system status occur, such as weapon malfunctions, personnel availability, etc. These inputs are then put through a data-processing system which contains a series of rules and procedures for manipulating the resulting data, using appropriate computing equipment to produce relevant outputs. These outputs are given in the form of reports or displays to appropriate managers in the control system. These may show weapon statuses, personnel shortages, equipment downtime, etc. Thus, a major function of the information system is to boil down or transform the very complex real system to a few elements that can be grasped and evaluated by a management group or organization. This evaluation leads to decisions which then result in management actions that change the future system status in a way that is intended to increase operational capability. Thus, the control system consists primarily of a set of policies or decision rules for achieving increasing operational capability, some of which are automated as part of the information system, an organization for decision making and taking management actions, and an information system.

IV. Difficulties in Designing Control Systems for the Future

In other words, an essential requirement in the design of control systems is the need to define the role of man and his relationship to the hardware. The difficulty in doing this arises from several circumstances. First, the control systems require several years to develop and produce, the most ambitious taking from 5 to 10 years. It is therefore necessary for the designer to project his requirements at least that far in advance. With the rapidly changing military technology, the environment within which the control systems must operate will be radically new and will create conditions that are very different from those existing today. Much higher alert requirements are likely, with a further need for fast response to critical situations, under conditions of wide dispersion. Radical weapons, including missiles, possibly space vehicles, and nuclear powered vehicles are in the offing.

Not only will the environment be very different, but because of the potentialities of new communicating hardware, such as data processing and communications equipment, and the developing research in organization and information theory, the designer finds it very difficult to project himself readily into this highly complex and different situation in order to take maximum advantage of these new factors in the design of the system. The relationships involved in these complex man-machine systems are often far too involved for intuitive or analytical treatment alone.

Although we can be sure that the environment of the next 5 to 10 years will be very different, we do not know the exact specifications of it. There are many uncertainties in the rate at which new weapons will be developed, both of an offensive and defensive nature, and the size and kind of enemy threat. These conditions as well as many others, can differ enough so that the demands on the men and machine in the control systems can only be estimated with uncertainty. However, the designer needs to know in detail the impact of this uncertainty upon the specifications of the control system so that the hardware manufacturers, the programmers, the training activities, etc. can carry out their assignments in a timely, coordinated, and comprehensive way. Thus, the core problem in the design of control systems is planning complex man-machine systems, requiring several years to produce, under conditions of uncertainty. In the remaining portion of this paper, we will discuss some of the general aspects of the design of control systems, and how simulation, as used in RAND's Logistics Systems Laboratory, can help in treating this core problem.

V. Use of Simulation

The definition of the elements of a control system in some detail produces the design of a feasible and compatible operational control system. The creation of this design is no simple task because it requires the designer to specify with some precision just how the decision maker in the operating system will act to increase his operational capability. The designer, therefore, has to visualize how the operating system would work. If he had an operating system to study, he could obtain such a visual aid, but the one he is designing will not exist for several years, and he cannot wait. Typically, in the past this lack of an operating system to study and manipulate has led to inappropriate, incomplete, and infeasible system designs. The gaps and defects were then left to actual operations to correct. This is both an inefficient and lengthy procedure which results in delays and difficulties in obtaining the maximum capability with the weapon system. The point of this paper is that if the operating system can be simulated in appropriate detail and under realistic conditions, the simulation can help fill the void at least partially, and perhaps, therefore, help produce better designs with resulting less stresses for the real world adjustment.

In recent work at the RAND Logistics Systems Laboratory, efforts were made to use simulation to help in this planning process and to design an operational control system. We reasoned that if we could simulate the environment of a future weapon system in sufficient detail we could then take the proposed hardware characteristics, operating plans, and support plans and determine their mutual compatibility, and, furthermore, that this would help us to estimate the operational capability they would produce for the weapon system. We used simulation because no real-world operating system existed, and yet we wanted to study an operating system early enough in the planning process to be able to help evaluate and possibly influence the proposed plans. Since we realized that our simulation of a future environment was subject to uncertainty and error, we varied it over a range of foreseeable conditions and hoped, in this way, to allow for our ignorance of the future. Second, we exposed the policies of the system to the range of simulated environments, and in this way determined their compatibility and overall performance. Third, we also used the simulated environment to help specify the details of the control system design, and by using this control system in conjunction with the other policies, we were able to determine how the particular control system which we employed modified the other policies, and therefore affected the performance effectiveness and resource requirements of the total system.

We believe that this study produced many useful results, both in the specific context of the weapon system we used, and in the techniques that evolved, which we believe have an application to many other such weapon systems, since each of these weapons is characterized by the planning difficulties we have described. We should like, therefore, in the remainder of this paper to describe the simulation experiment that helped to produce a design and an evaluation of an operational control system. This experiment is known as LP-II, and one of its primary objectives was the design and evaluation of a control system for an ICBM weapon system of the 1963-1965 time period. The control system that was developed is for a tactical unit, such as a squadron or wing, for two reasons: the ultimate effectiveness of the weapon system depends on performance of this unit, and second, the bulk of the resources and cost of the weapon system are also in the tactical unit.

VI. Description of Simulated System

Chart 3 contains a schematic diagram of the simulated system. The first requirement was to define the hardware of the weapon system under study. Engineering study and analysis gave us estimates of the missile configuration likely in the 1963-1965 time period. The long time required to convert hardware specifications into field equipment gave us some assurance that we had captured the essence of the missile hardware, and further, that for practical purposes it would be most realistic to assume that this part of the system was basically fixed. We found that we had to describe the hardware in what we called "Support Unit" terms in order to capture the hardware characteristics in sufficient detail to permit interaction of the operations and support activities in the tactical unit. A support unit was defined as a module, black box, or a functionally integrated component. For example, the checkout and malfunction detection equipment was basically designed to detect failures within support units, so that the support unit normally would be that module or component to be removed and replaced on the missile to correct a malfunction. To describe the missile, its ground support (launching and monitoring) equipment, and launch facilities in this level of detail, we found took 1,500 support units: 800 on the missile, 250 on the ground support equipment, and 450 in the facilities.

To represent adequately the operations-support interactions of this tactical unit in a minute-to-minute way, we found required about a hundred different parts characteristics for each support unit. These characteristics covered such elements as the skills and equipment needed to perform remove-and-replace activities; missile system checkouts; diagnosis of an ambiguous malfunction as reported by the checkout equipment; calibrations, and so forth. A very significant characteristic of each support unit was its reliability. Since this is one of the most uncertain parameters, its value was represented by a range from minimum-acceptable to maximum-likely reliability. In the course of the experiment, we examined the effect upon the control system of varying the reliability over this range.

A "failure" model had to be constructed which would be consistent with the reliability values, and which would define for each support unit the probability that it would fail under a given type of stress. We defined six types of stresses likely to cause failure for each of the 1,500 support units. The operational and maintenance statuses were defined for each of the 40 conditions in which the missile system could be placed: various degrees of alert, launching, countdown, checkout, replacement, etc. The description of each status defined the stresses that applied to each of the 1,500 support units by 15-minute intervals of time. Therefore, given a particular missile system status, the probability of failure for each support unit could be established for each 15-minute interval, and by making random draws from a probability distribution, a malfunction generation model was created. This model generated a significant portion of the workload for the squadron (identified by the dashed line in Chart 3).

The squadron contained several launch complexes and a squadron headquarters. A full squadron complement would normally contain several hundred people, but for the purposes of our study we were interested primarily in the management personnel. We identified the launch complex commander for each launch complex and the operations, maintenance, and supply officers of the squadron headquarters as being the key management personnel. Therefore, they were the "people" of the simulated squadron, and personnel to staff these six positions were brought to the laboratory from Strategic Air Command missile organizations.

This squadron organization was given a set of operational requirements, policies and resources by the embedded organizations who represented higher headquarters. The embedded organizations were staffed by laboratory personnel. These embedded policies specified the operations, maintenance, supply, manning, and data-processing requirements to be met by the squadron. The resources given to them were initially derived from Air Force manning documents, equipping documents, supply tables, etc., that were in effect at the start of the experiment. These resources were represented by punch cards, which could be assigned to jobs in the performance of operational or support activities by the managers of the squadron.

The squadron managers also had access to a data-processing and analysis center. One function of this center was to maintain the maintenance and supply-status data of the squadron, and to perform various logistical functions, such as automatic resupply notifications, required engineering changes, maintenance personnel and equipment availability and utilization, etc.

The experiment was operated by giving the squadron a set of operational requirements to meet, such as maintaining a specified amount of target coverage and a number of missiles on alert. The squadron was also given certain preventive maintenance and engineering tasks to perform. As the squadron managers tried to satisfy these operational and logistical requirements, they had to schedule the missiles to be in various situations or statuses. These statuses in turn caused stresses to be applied to the support units of the missile system, which caused malfunctions. The squadron managers then used the resources of the squadron to correct these malfunctions; and, throughout the entire exercise, the squadron tried to schedule its activities with the given resources so that maximum operational capability was attained.

As the experiment proceeded through several runs,* the operational and support policies were changed, the assumed reliabilities were varied, the resources provided were modified, and the organizational structure was altered. At the same time, the decisions made by the managers were studied, both by observation of the results of their decisions and by interviewers. Such analysis produced insights into the nature of the decision processes, and suggested possibly preferred decision rules. For some of these decision

*Each run lasted one simulated week. We found that such a time period was sufficient to permit most of the stochastic elements in the system to "settle down." A simulated week took about one week of real time, but time compression was achieved since we simulated the 24-hour days, 7-day weeks, during a real working week of about 35 hours.

processes, we used mathematical models or heuristic programs to develop the decision rules. The use of such rules in turn suggested required information and organizational arrangements for making decisions. This led to an evolving operational control system, which is the result of experience by Air Force people in the simulated environment plus analysis of the results obtained by various techniques tried during the experiment.

VII. Resulting Operational Control System

Referring to Chart 2, the control process illustrated is the classical one of information and feedback.* The novel problem in LP-II was to adapt this process to the military environment posed by ballistic missiles in the 1963-1965 time period. We can foresee a situation in which missiles are dispersed over a wide geographic area, maintained on a high level of alert, and responsive to sudden emergencies. The cost of these missile systems is very great, and so every effort will be made to keep the resources and costs they need to do their job to the minimum. One means of achieving this goal is to be able to shift resources (within the constraints of possible sudden war) from one place to another responsively, since the demands for resources at any one place are typically sporadic and low. Further, the support costs in equipment and facilities are so large a fraction of total system cost that minimum missile system downtime is desirable not only for military reasons, but also for economics, since support resource levels are almost fixed, in the short run, for a missile base. Since the cost of a weapon down is so great (if we value it at system cost per time period), the major problem for a tactical unit is to use its fixed resources to maximize alert.

This, then, is the environment in which the simulated control system had to function. The scheduling rules assigning missiles to various degrees of alert, and assigning the resources to repair missiles requiring preventive or other prescribed maintenance were found to be the critical decisions in the squadron. The formulation of precise scheduling rules by mathematical analysis was very difficult because of the many combinations of conditions and constraints that had to be considered. In effect, the implications of the schedule had to be determined by support units, and there were over 50,000 such support units on the operational missile system in a tactical unit. Further, the schedule was affected, to some extent, each 15 minutes as a missile or a resource changed status.

The useful scheduling rules have been developed by trying many rules of thumb during the experiment. The rules that seem to work best are called "opportunistic scheduling." Under this rule, all prescribed maintenance is deferred as long as the policies permit. Then, when a missile malfunctions, all possible prescribed maintenance as can be done concurrently is accomplished at that time. In this way, missiles are taken off alert as little as possible for prescribed maintenance.

Clearly, such a rule requires a highly responsive organization and information system because it must be ready to react instantaneously to any missile malfunction, change in resource status, etc., under conditions of wide dispersion.

The structure of the information system that has been designed to meet this requirement of unscheduled and rapid responsiveness contains the following: the current status of each missile and resource, a complete listing of all missile and maintenance situations and the resources required to accomplish them, and a listing of all prescribed maintenance awaiting performance on each missile. As a missile or resource changes status, the information system is updated through appropriate inputs. As a new situation occurs, each of the managers interrogates the information system for the implications it has for his function (such as operations, supply, or maintenance), revises his schedule accordingly, and assigns available resources to the urgent activities.

During the experiment, the information system was able to respond instantaneously to such interrogations. This was made feasible by placing all the necessary information in a very large random-access memory that could be interrogated very quickly. Since the system status changed very rapidly, an input system had to be devised which would quickly update the status, and this further required the minimizing of input data. This was done by requiring only a single input of each necessary data element, which served all managers.

Since the organization had to respond very quickly to emergencies, a good deal of functional integration was required. Operations, maintenance, and supply in the squadron headquarters had to work as a close team since each was affected by such emergencies. In addition, close communication between the launch complexes and the squadron headquarters had to be maintained. Report formats evolved which helped to present to each manager the consequences of another function's decisions on him, and the consequences of his decisions on them. In this way, each manager was able to react rapidly to decisions that might affect his plans.

Further, the squadron headquarters found it could not centrally handle all of the decisions, since many of them were localized to a particular launch complex. It therefore worked out conditions under which the launch complex could act without prior approval of the squadron headquarters, and when such prior approval would be necessary. These conditions have also been further developed by the laboratory staff into rules of thumb, since the relationships appear to be too complicated for analytic solution. Such heuristic techniques were

*RM-2131, Management Information for the Maintenance and Operation of the Strategic Missile Force, D. Stoller and R. Van Horn, The RAND Corporation, April 30, 1958, describes the requirements of management and information systems for a missile environment.

found to be very useful during the experiment.

The data produced by the squadron managers in the course of their minute-to-minute operational decision making were also used by them for longer-run or planning decisions. Thus, these data were used to estimate resource demands and utilization which provided the basis for determining required manning, equipping, and stockage of the launch complexes and squadron echelons. These data reflected the particular decision rules used to schedule the employment of the resources, and so there was a direct relationship between the short-run operational decision rules, like scheduling, and the longer-run planning decisions, like determination of the resource requirements of the squadron.

Thus, the simulation helped to produce and evaluate a control system that might fit the peculiar environment of the 1963-1965 missile era. The nature of the decisions, information system, and organization for such a control system has been developed in considerable detail, and as such it may provide a good guide to the design of such a system for the real world. Further, this blueprint is available early enough to permit its appearance in tactical units by 1963. We cannot say that this is an optimal control system, but it appears to be feasible and compatible with the type of environment experienced in the laboratory. Simulation thus has helped produce a reasonably detailed design of a control system for a future radically different environment.

VIII. Computers and the LP-II Experience

There are two aspects to computers and a simulation like LP-II: first, computers are used in the modelling process and runs; and, secondly, we learn something about the computer needs of the management system under study.

Only a few words need be devoted to the first aspect, here. In addition to standard punch card equipment, two general purpose digital computer systems were used: one with magnetic tapes capable of microsecond arithmatic speeds (in our case, an IBM 704 system) and a data-processing machine with relatively slow processing speeds but with a large disk memory capable of rapid access to any of several million characters (in our case, an IBM 305 RAMAC). In the laboratory, we have, occasionally, used the high-speed computer more or less "on-line" with the simulation -- men making decisions which are fed into the computer which in turn quickly "plays the decisions" through various models and returns to the relevant men the system impacts of these decisions, which in turn causes new decisions, etc. But in LP-II, most of our operations on the high speed computer were not "on-line". They were either for analysis purposes or for computing lists of "potential failure rates" which were then interpreted by laboratory personnel to reflect the minute-to-minute decisions of the laboratory participants (see the description of the failure model above). As a result, the entire project consumed only a few hundred IBM 704 hours, a relatively small portion of the total cost of the operation, which also required more than 70 man-years. In other manned simulations where the participants and the high-speed computer "talked" to one another many times daily, the computer costs were a considerably larger proportion of the project's total costs.[1]

The computer with the large memory (the RAMAC) on the other hand, was used primarily <u>on-line</u>, minute-to-minute, during the runs, and served mainly[2] as an information storage and interrogation device for the laboratory participants. Study of this computers' programs and use permitted us the statements we could make about the kinds of computers such a tactical unit might need if it were to obtain the same effectiveness as our simulated unit: The RAMAC operation was basically a "status of resources" system and served to provide the resource manager with "up-to-the-minute" data needed to make decisions on the assignment of these resources. Hence, all of its internal files were "status files" in the areas of maintenance, supply and operations. The inputs to the system were changes in status (personnel movements, maintenance actions, etc.). The outputs were status reports (generated in response to the manager's inquiries) and a comprehensive set of history cards. The history cards served as a historical file, and were also used to generate certain summary and history reports. The system had two basic components; the disc files which were the information acted upon, and the stored programs which did the acting.[3]

The system contained both "on-line" and "off-line" programs. Off-line programs operated independently of each other and of any central control. They were used to perform functions of an occasional, isolated nature such as loading the initial files. To use an off-line program, it was necessary to interrupt the normal 305 operative mode. On-line programs were all controlled by a Master Routine. These were the programs which processed the normal input cards. All on-line programs (and the Master Routine) were stored in the disc files. The Master Routine was normally on the drum and operating, and the processing routines were read in and operated as needed. Several of the on-line programs used subroutines. The subroutines were stored on the disc files and were read in and controlled by the programs which used them.

[1] We have been able to do this several times during the course of an 8-hour day, which simulated several daily or weekly "time periods".

[2] It was, also, invaluable for analysis purposes, later, following the runs.

[3] The remainder of this section is largely due to the efforts of J. Tupac and K. Labiner. See their RAND Research Memorandum, RM-2572, <u>The LP-II Data Processing System</u>, September, 1960.

The LP-II data system did not unduly tax the machine's storage capacity, using less than 90% at peak usage.[1] However, the machine's slow processing and output speeds did influence the system's performance considerably, and suggested the infeasibility of doing all that had been hoped for with little or no information lag. To quote from the report on the LP-II data system:[2]

'Slow processing and output speeds strongly influenced LP-II's system design. As mentioned, almost all machine time was used in processing the "on-line" status changes and interrogations. As a result, we observed three consequences. First, the machine did not generate all the summary reports. Had we used a faster machine, all summary reports could have been generated in step with its other operations and thus furnished managers with some summary data on an inquiry basis. Greater direct-printing capacity and speed would also have been required. Secondly, the machine did not make decisions. We found it was difficult to define decision rules when the system was being designed. In addition, a machine with much higher internal processing speed and computing capability than that available for the experiment would be necessary in order to incorporate decision functions. This would have been true even if the rules could have been established at the outset. Finally, there was not as much internal checking of data as one would have liked."

To summarize the major outputs of LP-II with respect to control system design are as follows:

1. An integrated control system for a large missile tactical organization, combining the organization, policies, data system, with the novel operational and weapon system environment of future years.

2. Estimates of workloads, resource utilization, delays, missile alert performance, etc., resulting from the interactions of environment, policies and decision-making of the organization.

3. Frequency of, and kinds of techniques used by the management personnel for making critical decisions, such as targeting, scheduling of missile status, and resource assignment, as tailored to the special demands of the large tactical organizations.

4. The information requirements for making critical decisions and performing management activity. The reports requested by the participants changed considerably as the organization was exposed to different situations, such as higher alerts, engineering changes, reduced resources and revised requirements thus generated provided an explicit estimate of the data displays that the new organization may require.

5. A detailed description of the data processing system for providing the required information. This system was based upon several new concepts that imply greater automation, new methods of reporting data, and a different type of data organization. These concepts were made operational in the simulation so that many of the specific techniques used may be directly transferable to a real-world control system. Also, estimates were obtained for the frequency and urgency with which different data elements were required. These may be useful in developing specifications for data processing hardware.

IX. Limitations and Future Research

The number of alternatives facing the designer of the information portion of a management control system are enormous, particularly during the planning stages. Chart 4 gives some idea of the number and kinds of choices that a system designer must resolve. For a given budget, he is expected to choose the right man-machine resource mix for each of the alternatives mentioned. Anyone who has ever helped design an information system can supply examples of the kinds of choices that Chart 4 represent.

We have said, so far, that manned simulation, properly combined with all-computer simulation models, can help the designer make these choices. LP-II, if nothing else, should increase the designer's belief that this is a possibility. However, lots of problems which remain must be satisfactorily resolved before this becomes part of the everyday tool-kit of a system design program.

1. The presently high costs of a simulation like LP-II. The sizeable scope, great detail and the use of humans in simulations are all factors which make this sort of experiment useable, primarily, for designing larger, complex "man-machine" systems. Of course, some of these costs would have to be borne by any sizeable study of a control system but others can be reduced by turning to suitable compiler programs, and using only the degree of detail necessary to obtain the desired results.

2. Experimental Design Problems. Manned simulation experiments cannot produce, for the same research budget, the same number and length of runs that their all-computer brethren can. This makes it difficult to perform the desired amount of sensitivity testing or run long enough to wipe out, fully, the effects of initial conditions. Work is proceeding at RAND, and elsewhere, on experimental design techniques to mitigate these effects.

3. Inability to Produce Optimal Designs. Simulations, by their very nature, produce at best highly preferred -- but not optimal -- policies. Significant betterment in management control design is, of course, nothing to belittle. The development of all-computer simulation and analytic

[1]This was partially due to the scale used in the simulation models, however.

[2]RM-2572, op. cit., pp. 37-38.

models based on the manned simulation as a breadboard model should help, considerably, to produce even better results.

4. Programming Task Synchronization. Coordinating the many tasks through time that have to be performed by the computer portion of the simulation has, in the past, ordinarily required setting the basic time unit of the simulation equal to the minimum time unit of all tasks. The possibility of using variable time synchronization -- different tasks running at different time units -- is being investigated. This would permit running the simulation activity at differential speeds as a function of the question the particular simulation run is addressing.[1]

5. Defining a Suitable Level of Scope and Detail. The system designer might wish the simulation activity to produce answers to highly specific questions but this requires the simulation activity to represent great amounts of detail and many functions and organizations. This is costly and makes analysis more difficult. Therefore, we must better understand how to deal with management control problems in an abstract way for simulation purposes, yet in a realistic enough manner to produce valid design factors for real-world application.

X. Conclusion

Notwithstanding these difficulties, we nevertheless believe that manned simulation will become a very useful technique for those seeking to make choices in a system context, between alternative management control resource mixes.

The problem described in this paper is probably characteristic of many large system-design problems. Certainly the need for such operational control systems is expanding in the military, and in both civilian and military the need for improved management systems has been ever present. Simulation techniques can prove of much help in these problems by providing a means of pooling and integrating knowledge from many sources and by providing the opportunity to iterate and vary the many variables and parameters that compose such systems. Although most published simulation experiences have involved all-machine models, we have found much value in man-machine simulation when the problems have involved organizational interactions, the design of information systems, and conflicting or interacting decision rules, since these undergo considerable development during the simulation process.

[1]Jack Little of RAND's Computer Science Department is working on this problem.

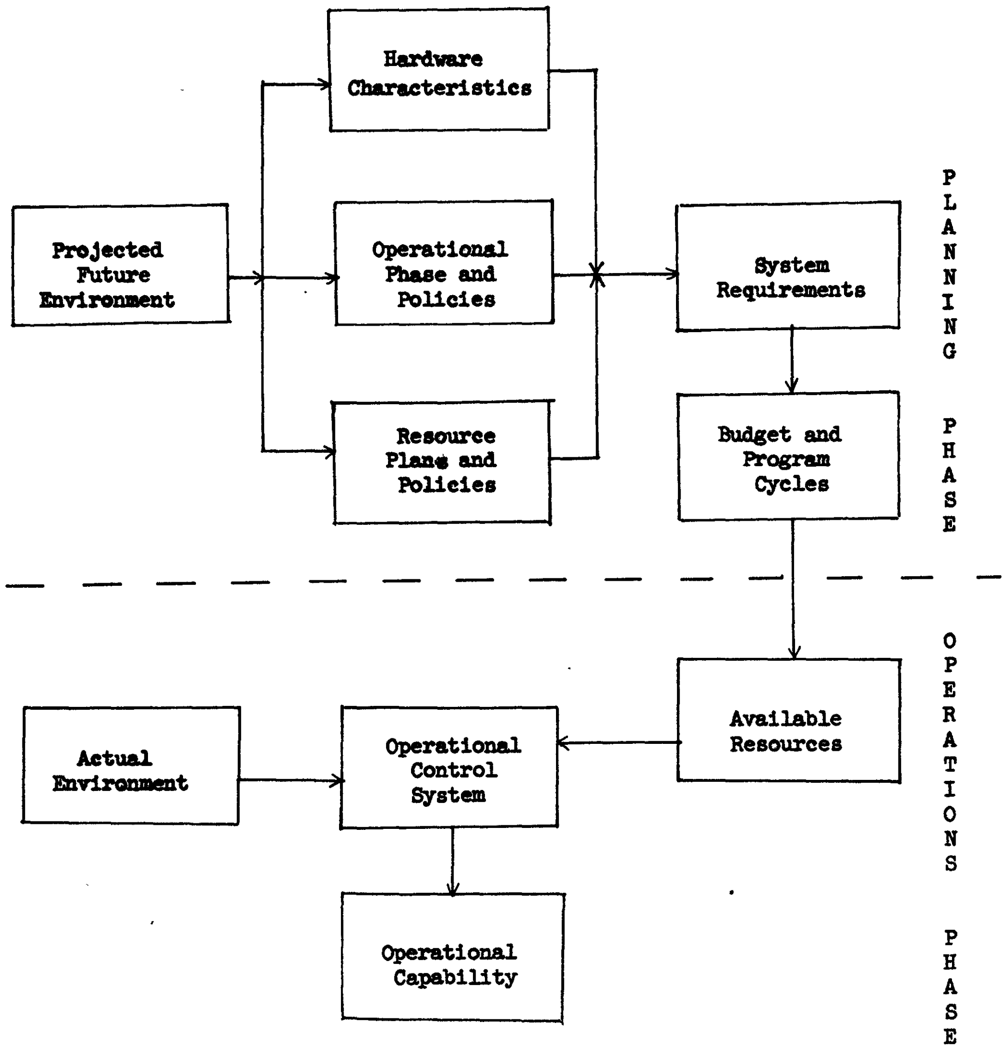

Hardware Characteristics
Projected Future Environment
Operational Phase and Policies
Resource Plans and Policies
System Requirements
Budget and Program Cycles
PLANNING PHASE
Actual Environment
Operational Control System
Available Resources
Operational Capability
OPERATIONS PHASE

Chart 1

PLANNING-OPERATIONS RELATIONSHIPS FOR A FUTURE WEAPON SYSTEM

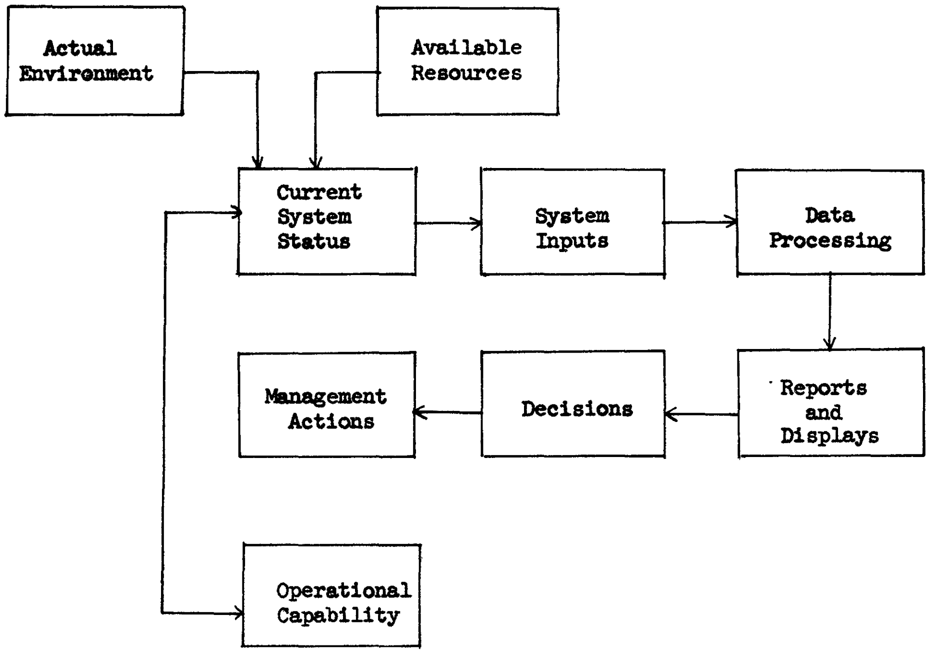


Chart 2

ELEMENTS OF AN OPERATIONAL CONTROL SYSTEM

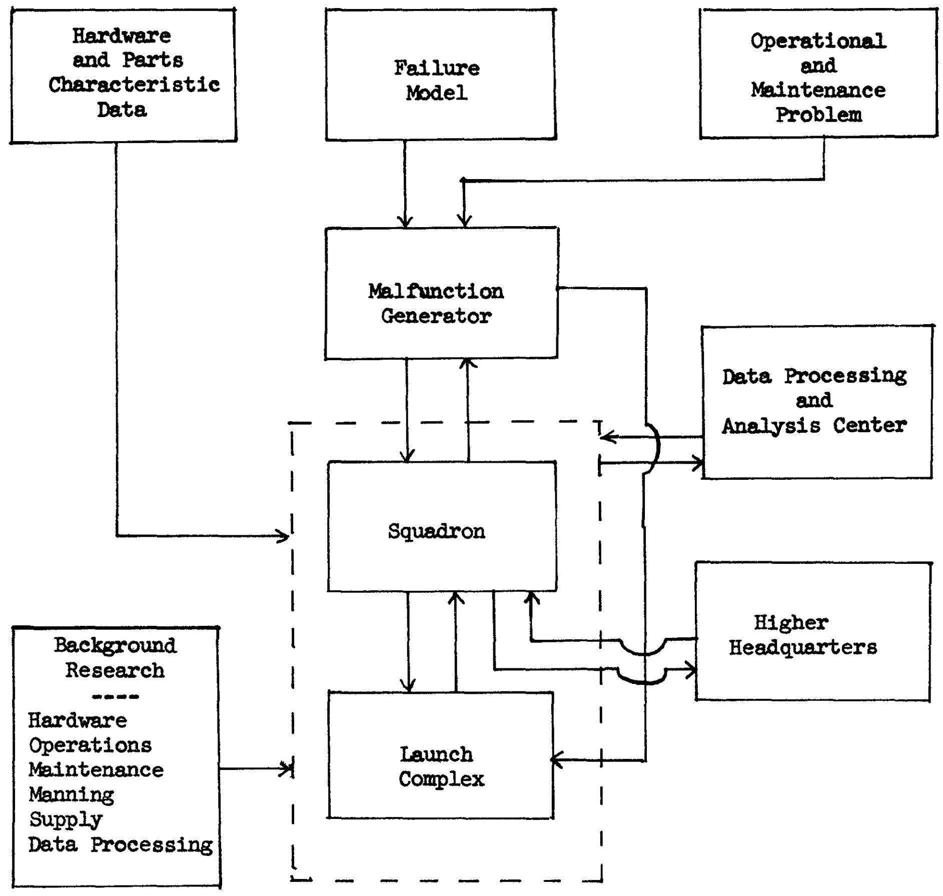
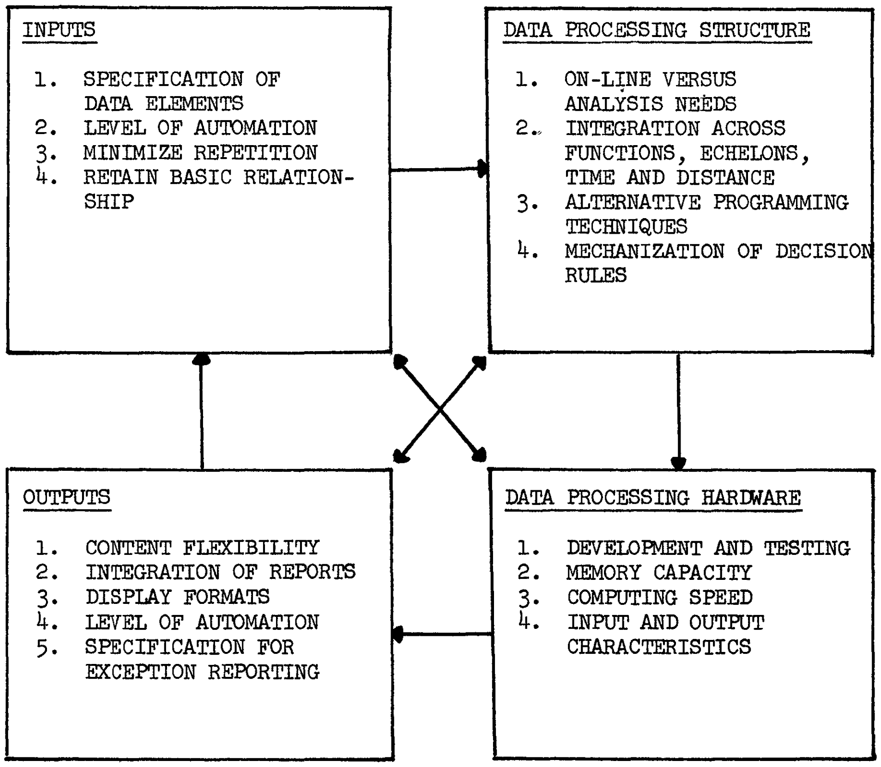


Chart 3

MODEL OF SIMULATED WEAPON SYSTEM

CHART 4

FEATURES OF ASSUMED INFORMATION SYSTEM DEVELOPMENT


INPUTS
1. SPECIFICATION OF DATA ELEMENTS
2. LEVEL OF AUTOMATION
3. MINIMIZE REPETITION
4. RETAIN BASIC RELATIONSHIP
DATA PROCESSING STRUCTURE
1. ON-LINE VERSUS ANALYSIS NEEDS
2. INTEGRATION ACROSS FUNCTIONS, ECHELONS, TIME AND DISTANCE
3. ALTERNATIVE PROGRAMMING TECHNIQUES
4. MECHANIZATION OF DECISION RULES
OUTPUTS
1. CONTENT FLEXIBILITY
2. INTEGRATION OF REPORTS
3. DISPLAY FORMATS
4. LEVEL OF AUTOMATION
5. SPECIFICATION FOR EXCEPTION REPORTING
DATA PROCESSING HARDWARE
1. DEVELOPMENT AND TESTING
2. MEMORY CAPACITY
3. COMPUTING SPEED
4. INPUT AND OUTPUT CHARACTERISTICS

A SURVEY OF MICROSYSTEM ELECTRONICS

Peter B. Myers
Motorola Inc., Semiconductor Products Division
Phoenix, Arizona

Summary

The history of microsystem electronics is briefly traced through the successive stages of: miniaturization; subminiaturization; microminiaturization; thin-film integrated circuits; semiconductor integrated circuits; and finally morphological integrated circuits or functional blocks. The term integrated circuit is defined as the combination, on or within a single chunk of material, of multiple electrical elements to perform a desired circuit or systems function. Improvement in reliability, decrease in size and weight, decreased power consumption, and lower cost are discussed as motivations for Integrated Circuitry. Fabrication techniques, interconnection, and accessibility are identified as major problems. A catalogue of techniques for fabrication of Integrated Circuitry is presented.

What Is Microsystem Electronics?

We define Microsystem Electronics as that entire body of electronic art which is connected with or applies to the realization of extremely small systems. As such it includes power supplies, input and output transducers, and various forms of memory where applicable, as well as digital and analog electronic circuitry.

This paper surveys the techniques of microsystem electronics that apply to the arithmetic or logical elements of computers and to the electronic circuitry of communications equipment. The electronics industry has not yet agreed on a name for this part of microsystem electronics but the term Integrated Circuitry seems to have wide and growing acceptance as a generic name.

Integrated Circuitry

Definition

We define an integrated circuit as a combination, on or within a single chunk of material, of a number of basic electrical elements to perform a circuit function.

The concept and terms used can be illustrated by a simple four-box picture of electronic technology. Let us arbitrarily divide electronic technology into the following four mutually exclusive and exhaustive categories: Materials, Components, Circuits, and Systems. If we investigate the emissive properties of a heated tungsten wire or the dielectric constant of polyethylene we are clearly studying the properties of Materials. If we combine a tungsten wire, some bits of nickel wire and sheet, put it all in a glass envelope which we then evacuate, we have made a vacuum tube -- a Component. If we properly connect our vacuum tube, an inductor, a capacitor and a battery, we have an oscillator -- a Circuit. Finally, if we combine an oscillator, a modulator, an amplifier, a power supply, and an antenna, we have a transmitter -- a System.

Note that in the above example we were engaged in four distinctly different types of activity. In the specialization of today's electronic industry these activities are performed by four different men. In the example, and more generally in our four-box picture of electronic technology, we find communication and interaction -- bi-lateral feedback -- between each type of activity and its immediate neighbors. The components man, for example, takes the output of the materials man, asks him for materials with specific characteristics, and tries to feed back the results of his experiements in such a way that the materials man can modify or optimize his output. The components man also communicates with the circuits man, listening to his requirements and providing or designing components to fit. He receives feedback from the circuit designer as to how well his components work and how they should be changed.

Similarly, the circuits man goes to the components man for his building-blocks and to the systems man about the finished product he hopes to supply. Each man, in addition to his own problems and language, must be able to communicate with his neighbors sufficiently well to exchange the information necessary for his work.

Note, however, that in this simple picture there is no need for the materials man to talk to the circuits man or to the systems man, and in practice he hardly ever does. In fact, they have so little in common in our conventional electronic technology that they don't really speak the same language. Likewise the components man has no real need to understand the problems of the systems man, and the latter tends to think of components only as black boxes with certain failure rates which limit the reliability of his system.

This partitioning is a result of the need for specialization in today's technology and of the finite amount of time and interest the average technical man has for exploring outside his own speciality. It is unfortunate in that it leads to inefficiency and lost opportunity in a growing number of instances where a new requirement is met by an adaptation of an old technique instead of by a fresh examination of the wide range of unexploited possibilities.

Integrated Circuitry removes the internal partitions and integrates the boxes into a single field. Integrated Circuitry still has its specialists, but each must have a working knowledge of the problems and objectives of areas which yesterday would have been the exclusive domain of other specialists. For example, the circuits man may no longer limit his consideration to the black box characteristics of existing components. He must be aware of the characteristics of the materials which go into his components, of the limitations imposed by the laws of the solid state, and of the possibilities it opens up. The components man may no longer accept blindly the requirements set by the circuit designer. He has a responsibility to examine the overall requirements on the system to find the most satisfactory way of accomplishing the desired function. As an example, a conventional circuit designer might build a power supply with a transformer, a diode, a choke, a couple of capacitors -- or a simple RC filter, if you prefer -- in either case he has used a minimum of five components. Our integrated circuit designer might choose to build the same power supply from a single piece of silicon. Alternating current flowing through one part of the silicon encounters resistance and generates heat which filters through the silicon to a thermoelectric area, where Seebeck Effect produces filtered direct current. This particular example happens to be one in which the Integrated Circuit bears little resemblance to its conventional counterpart. The basic electrical elements used include resistance for the generation of heat, transistance in the conversion by Seebeck Effect of heat to direct current, and thermal and electrical insulation at appropriate places.

Basic Electrical Elements

New as it is, Integrated Circuitry is made up of the same six basic electrical elements which long have been the ingredients of the electronics art. They are insulation, conduction, resistance, capacitance, inductance, transistance, and some special combinations of these. By insulation we simply mean the prevention of current flow, or the isolation of electric or magnetic fields. Conduction signifies the free flow of electric current and resistance, capacitance, and inductance have their customary meanings. Transistance is a new and highly useful term which describes the gain of active elements, or their ability to achieve precise control. It applies to all conventional forms of transistors, diodes, and other Solid State active elements. Finally, by special combinations we mean elements which can only be approximated by networks in our conventional technology. The prime example is the distributed resistance-capacitance network realized in Integrated Circuitry by a thin-film resistance deposited upon a dielectric layer, which in turn has been deposited on a conducting medium.

Evolution of Microsystem Electronics

To gain perspective in our survey of microsystem electronics let us look at its history and evolution.

The trend was set by miniaturization which made use of smaller forms of conventional components interconnected by conventional wires and solder. In every case the discrete nature of the individual component and usually its conventional form has been maintained. Miniaturization represents the first step, chronologically, in the attempt to make electronic equipment smaller.

The next step was subminiaturization with still smaller forms of conventional components. In most cases the conventional form of the component has been maintained but the size and weight are reduced to the point at which thin wire leads provide adequate support for mounting. The cordwood technique, in which cylindrical axial-lead components are stacked like cordwood with their leads fed through matched holes in parallel printed circuit boards in front of and behind the stack, is a good example. Various forms of printed or etched circuitry, both rigid and flexible, are used as the combination wiring harness and support.

Microminiaturization is the name given to the ultimate size reduction of the individual component. It differs from subminiaturization in that the conventional shape and form factor are generally lost, leads are often left off, and a supporting board or matrix is always necessary. Microminiaturization still permits the circuit designer uninhibited freedom of choice in the selection of his individual components.

Thin-film Integrated Circuitry represented a major advance by dispensing with separate mechanical supports for each component and combining multiple thin-film components on a single glass or ceramic substrate. Overlapping or touching films form internal connections. At present a hybrid form of thin-film Integrated Circuitry is necessary since none of the many companies working in this field has perfected a way to fabricate workable thin-film diodes and transistors on a glass or ceramic substrate. Current thin-film integrated circuit technology involves deposition of thin-film passive elements followed by the applique of any required active semiconductor elements.

Semiconductor Integrated Circuitry means the combination of thin-film and semiconductor circuit elements on and within a single crystal semiconductor substrate. Connections are made by deposited thin-film conductors, and by physical juxtaposition. Of all the existing forms of Integrated Circuitry, the Semiconductor version permits the widest variety of active and passive elements and the greatest potential flexibility. Active elements can be formed within or on the substrate where needed, and either thin-film or semiconductor techniques can be used to form the passive circuit elements. When semiconductor technologists have perfected a means of depositing thin-film semiconductor active elements on glass or ceramic substrates, the passive substrate approach may well prove more flexible than the active substrate approach. This comes about because part of each active element is unalterably connected to a common semiconductor crystal in the active case. Although careful placement of active elements and tailoring of the shape and thickness of the active substrate between may allow quite complicated circuits to be built into a single semiconductor substrate, more complex circuits could undoubtedly be achieved if both active and passive elements could be deposited on an insulating substrate.

Morphological Integrated Circuitry or what some people call the functional block, represents the combination of solid state materials to perform a desired circuit function, although neither individual components nor precise electrical circuits are necessarily identifiable. The power supply mentioned earlier is a Morphological Integrated Circuit. Another example is the familiar standard frequency quartz crystal, which is a homogeneous slab of material although it acts as a combination of resistance, capacitance, and inductance. The lack of a physical counterpart in conventional circuitry makes this type of Integrated Circuit the most difficult to design. However, there is a long list of unexplored physical effects awaiting our attention, and the possibilities are unlimited.

In the rest of this paper we shall limit our consideration to Integrated Circuitry, as defined above, since it represents the most significant departure from conventional technology. The following sections first review some of the reasons for developing Integrated Circuitry and then present a catalogue of the basic applicable fabrication techniques.

Philosophy of Integrated Circuitry Development

Having briefly examined what it is and the evolutionary stages through which it is developing, let us consider some of the motivations for the development of Integrated Circuitry. Although there may be disagreement as to the order of importance, it is not hard to identify the following areas of concern as having motivated the electronics industry in this field: reliability, size and weight, speed, power consumption, accessibility and cost.

Reliability

Everyone talks about reliability, and it is certainly true that without a definite minimum reliability neither microsystem nor any other kind of electronics can long endure, but is there any significant reason to expect a better reliability from Integrated Circuitry than from conventional component electronic circuitry? Yes, say its proponents, for at least two reasons: first, because the number of discrete point connections is greatly reduced and connections have always been one of the sources of failure; and second, because the overall reliability of the system is no longer the same complicated function of the individual reliabilities of each individual component. This latter claim stems from the common mode of fabrication of Integrated Circuits where usually all of a given type of electrical element, say capacitor, are created at the same time. This common fabrication may seem to have more bearing on process yield, and hence on cost, than on reliability but there is a connection with reliability buried back in the reduction of total process steps and the consequent greater attention that must be paid to each process step.

It is undeniably true that our electronic systems, be they computer, communications, or weapons control, are getting progressively more complex. With increased numbers of components all having to function together, we seem to be approaching an asymptotic barrier where additional complexity can only be achieved at the cost of decreased reliability. While we are still a long way from the machine with infinite complexity and zero reliability, our modern systems are already uncomfortably close to the complexibility-reliability barrier.

Size and Weight

At least three sources of concern can be grouped under the heading of size and weight. For missile and other airborne applications the size and, often more important, the weight of an electronic system has a cost in propulsion machinery that provides a strong motivation for reducing both. While reliability is of very great importance in some of these applications, reducing size and weight to particular values can mean the difference between the possible and the impossible.

A second size and weight motivation is found in the realm of portable products where the size of a potential market, both industrial and military, often depends on whether a system can be created and packaged within certain limits.

A third motivation for size reduction is the economic, where an equipment must be housed or protected or otherwise maintained in certain conditions. If the cost of providing a required environment is high enough and is proportional to the volume required, considerable effort can be justified in reducing the volume. A final motivation is shared with the next area of concern.

Speed

For a long time increasing the speed of electronic systems was a matter of improving the components. Recently the speed of certain systems has reached a point where propagation delays in the wiring, rather than the characteristics of the devices, prohibit faster operation. This point is often called the speed-size ceiling and it has important bearing on computer systems. If a particular system has reached its speed-size ceiling, it is impossible to expand it functionally without decreasing its speed of operation, and vice-versa. The only way through a given speed-size ceiling is to decrease the physical size of a given system and thus shorten the propagation delay through it. Microsystem electronics, and particularly Integrated Circuitry, shows promise of substantial improvement in speed-complexity product by significant reduction of propagation delay within and between integrated circuits.

Power Consumption

Information, in an electronic system, must be related to some form of energy if it is to be processed. In general the amount of energy contained in a given piece of information is very small in comparison to the energy expended in processing it. In part this is because of the inefficiency of the processing equipment, but often a much larger drain is deliberately introduced in order to assure continued operation in the event of drift or change in certain component characteristics. Where many different processes are involved in fabricating the components for an equipment, many different and often uncorrelated drifts and changes may be expected. To assure proper operation under these conditions may require ten or one-hundred times the power needed in the absence of change. The advantage of Integrated Circuitry in this respect is based on the reduction in number of different processes involved in fabricating the equipment. If all resistors have the same temperature coefficient and aging characteristic, for example, variations in bias and operating point of associated active elements will be much reduced.

As we shall show in the catalogue of integrated circuit techniques which forms the major part of this paper, not all resistors useful to Integrated Circuitry are made by the same process. However, many of the resistors in a given integrated circuit will not only be made by the same type of process but will also have been made at the same time from the same material and under a single set of conditions.

Accessibility

There are a number of facets to the problem of accessibility of Integrated Circuitry. First, it is apparent that those parts of electronics systems that must interact with a human operator must be matched to him in physical size. Knobs must be such that fingers can grasp them, dials of a size that the eye can read them, and so forth. Input and output equipment must be matched to the environment from which it receives and to which it gives information. These are problems of accessibility where limits of useful size reduction can be determined and beyond which it is neither reasonable nor desirable to go.

The second set of problems revolve around the need for replacing defective parts of the system upon failure. It is no longer feasible to think of "repairing" a failure. When some part of an integrated circuit as we have defined it stops working, it cannot be operated upon in the field. The service man must treat the entire circuit, perhaps even a group of related integrated circuits, as a single "component" which he replaces by a new "component" from an inventory of known working spares. But we cannot afford to throw the radio away just because the dial cord is broken, and hence we have the problems of suitable modular subdivision and of accessibility for replacement. The problem further breaks down into one of size and one of interconnection.

The problem of size in accessibility for replacement is much like the first problem of matching to a human operator. The technician must be able to get a grip on the faulty module to remove it.

The problem of interconnection is much deeper and more fundamental. It reaches across and can nullify the motivations of reliability, size and weight, and speed as well as accessibility. Many present day electronic systems use almost as much volume for interconnecting wiring behind the panel as they use for components out in front. Unless the connecting means from circuit modules to wiring harness is more reliable than the Integrated Circuitry itself, system reliability will be hurt. If the wiring harness is physically long, the propagation delay in getting the signal from one part of the system to another reduces the maximum speed of operation. Finally, unless the interconnecting means is flexible and cleverly arranged, it may be almost impossible to get at the connections to replace a module, since the motivations of reliability and size tend to rule out the use of plugs and sockets.

The final set of problems of accessibility are concerned with the removal of heat. The potential saving in power through an allowable decrease in operating margins has already been mentioned, but even so a significant part of integrated circuit design is thermal in nature.

Cost

The cost picture for Integrated Circuitry is a complicated one. Initially some of the specialized applications for which there are no other possibilities will probably support Integrated Circuitry regardless of cost. Any sort of general acceptance, however, will require a cost competitive with other forms of circuitry. Since individual components can no longer be selected after manufacture, both control of process and process yield will have to be high. An aid in this respect is the fact that the same process run usually forms all of a given type of circuit element on a particular integrated circuit substrate and thus if one part is good all tend to be good. This simultaneous fabrication of all diffused resistance regions or all deposited capacitors at the same time is one of the advantages by which its proponents hope to lower the cost of Integrated Circuitry.

Catalogue of Integrated Circuit Techniques

Our survey of microsystem electronics may logically end with a catalogue of existing and proposed means for obtaining the elementary circuit functions. Since the assembly of complex circuit functions called for in a designable Integrated Circuit Technology must rest fundamentally on the precision and process control attainable in the basic circuit functions, major continuing effort has been put on the study of means for obtaining greater control over these individual processes.

The ten subdivisions which appear below are the ten basic elements needed by almost every circuit regardless of techniques used to build it.

Method of Catalogue

There are a number of ways in which Integrated Circuit Technology may be catalogued. We may catalogue it by process, by materials used, by physical form or arrangement, by the basic circuit parameters, by the energy forms and transformations, by generic circuit function and, finally, by specific circuit function in a representative system. For simplicity we have chosen to subdivide by basic circuit parameters:

1. Insulation
2. Conduction
3. Resistance

4. Capacitance
5. Inductance
6. Special Networks
7. Active Elements and Substrates
8. Encapsulation
9. Mechanics
10. Design.

The above catalogue is by basic circuit parameters rather than one of the other means of classification because for a survey such a breakdown seems to be most meaningful and concrete.

In developing these basic elements we sometimes use the bulk properties of the body of the substrate, sometimes operate at or close to the surface by alloying or diffusion, and sometimes build the element on top of the substrate by epitaxial growth of semiconductor materials, evaporation, plating, or other means of deposition of thin films of material. The mechanical, thermal, electrical, and chemical interactions resulting from these new approaches to electronic circuits have introduced both new problems and indeed a new science.

Insulation

Electrical insulation is required in all but the most trivial Integrated Circuits. Extrinsic techniques of insulation include operations external to the substrate. The development of extrinsic electrical insulation includes anodization, vapor phase deposition of dielectrics by pyrolysis, evaporation, and plasma deposition techniques as well as the conventional mechanical coating. Intrinsic insulation includes any method of isolating fields or current flow within a semiconductor or other substrate. Among these are thermal oxidation, fabrication of an isolating layer of intrinsic semiconductor material, and creation of one or more back-biased junctions.

Anodization is the formation of an insulating oxide over certain elements, usually metals, by electrolytic action. The most commonly anodized materials are tantalum, aluminum, titanium, and niobium. Anodization is a particularly useful form of insulation where protection of a conductor is required since the base metal can form the conductor and the anodized surface layer can form the insulator. Since anodized films of tantalum can be controlled to a high degree and possess a dielectric constant of approximately 25, the process finds wide application in formation of capacitor dielectrics. The high dielectric constant becomes a disadvantage as the frequency of the energy to be insulated increases.

Pyrolysis as used here is the thermal decomposition of a volatile chemical compound into nonvolatile and volatile byproducts. It is generally carried out in an inert carrier gas and this fact distinguishes it from vapor plating, which generally uses an active carrier gas such as hydrogen or steam. Silica, silica-based glasses, and silicone polymers have been deposited successfully.

Evaporation is the deposition in high vacuum of insulation thermally liberated from a parent source. Silica films of low optical absorption have been produced by electron bombardment of the parent oxide. The technique has important possibilities in the area of direct or mechanically masked insulation deposition.

Plasma Deposition involves the spraying of highly excited atomic particles of the insulator. Heat, commonly obtained by electric arc or hydrogen flame, is the source of excitation and almost any elementary material can be applied to almost any surface by the method. The disadvantages are the grossness of the spray process and the high temperatures involved.

Mechanical Coating covers the spraying, painting, or other physical application of organic and inorganic insulators and finds certain applications in the grosser insulation of Integrated Circuitry.

By means of extrinsic techniques broad area dielectric film insulation has been developed with field strengths in excess of 10^6 volts/cm. These insulating films can be reproducibly obtained to thicknesses in excess of 10 microns.

Thermal Oxidation is here used to denote the formation of a self-oxide upon the exposed surfaces of a semiconductor. The intrinsic insulation of silicon has been developed to a high degree by this method. Oxidation of parent material can be used to insulate broad areas and also PN junctions.

Intrinsic Layering refers to the method of separating two regions of conductive semiconductor by a region of near intrinsic semiconductor material which differs enough in resistivity from the adjacent regions to serve as an insulator. The epitaxial growth of silicon is a promising method of providing layers of intrinsic material.

Back-biased Junctions may be formed within semiconductor substrates to provide a type of insulation. With reasonable values of reverse bias the capacitance across a graded junction can be reduced to the point of insignificance for low-frequency applications. The division of semiconducting substrates into two or more regions isolated by reverse-biased junctions opens the way to an eventual increase in the number or complexity of the functions that might be built into a single substrate. A disadvantage of the technique is the collection of any minority carriers in the area.

Conduction

Conduction is used here to indicate electron current on or in substantially ohmic material of negligible resistance. One of the goals of Integrated Circuitry is to achieve integration of circuit functions on and within substrates so that ohmic connections from one location on the substrate to another are drastically reduced or eliminated. In cases where this is impossible or impractical, extrinsic conducting paths are formed by evaporation of metals, by painting or plating of conducting stripes, by sputtering, by pyrolysis or by mechanical coating. Intrinsic conduction can be accomplished by development of degenerate regions within the semiconducting or substitute substrate. We can do this by creating alloyed, epitaxial, or melt grown regions during or following crystal growth.

Evaporation of conductors covers the vaporization of metals at high temperatures in vacuums of 10^{-4}mm Hg or better. The metallic vapor moves in substantially straight lines onto a substrate which may be mechanically masked to limit the deposition to desired regions. The substrate must be very clean and must generally be raised to an elevated temperature in order to assure intimate contact of the particles with the substrate after impact before solidifying. Alternative means of producing conductive patterns involve coating an entire surface and subsequently removing unwanted conductive material by one of a number of techniques. Evaporation may be of a single metal or of an alloy. In the latter case the deposition may be made simultaneously from a common source or sequentially from separate sources, after which the substrate may be heated to produce an alloy on the surface of the substrate. If the alloy penetrates the substrate, as in the case of a semiconducting substrate, the result is classed as intrinsic.

In the study of conductors by deposited techniques, resistivities below one order of magnitude higher than metal conduction have been achieved. Where mechanical or thermal considerations make adhesion more important than achieving the lowest ohmic resistance the introduction of a chrome-gold alloy has been useful.

Sputtering differs from the evaporation process discussed above in the conditions under which the conductive material reaches the substrate. Sputtering is the result of a glow discharge between an inert anode and a bombarded cathode of the desired conducting material. Because the presence of gas at 10^{-2} to 10^{-4}mm Hg is necessary for the generation of the ionized bombarding molecules, the sputtering process is inherently harder to keep clean. Even so the process has certain advantages over vacuum evaporation in the relatively low temperatures needed or generated in the system, and thus the lower chance of contamination from the evaporative source. It finds one of its most important applications in the production of thin films of tantalum for resistors or capacitors.

Pyrolysis as used for deposition of conducting films is the same process discussed previously for insulators. It has the advantage of flexibility of materials and conditions of deposition but the disadvantage that the deposited material cannot be masked by mechanical shields as satisfactorily as the vacuum evaporated materials. Pyrolysis finds application where an entire surface can be coated as for electrostatic or magnetic shielding, or where the unwanted material can be removed selectively after deposition. Pyrolysis may also be carried out on selective regions under certain circumstances by employing a catalyst.

Plating of conductors includes electroplating, chemical or electroless plating, and vapor plating. Electroless plating tends to cover everything as does vapor plating. Selective removal of unwanted material can be combined with electroplating to build up conducting paths. Both electroplating and electroless plating suffer from danger of contamination from the wet chemistry involved. Vapor plating is capable of achieving high-purity deposition.

Mechanical Coating includes conducting glass pastes which can be painted onto gross terminal pads to bridge irregularities that vapor deposition cannot manage. It also includes various solders that may find temporary use in making conductive paths to external terminals. The conductive glasses have the advantages of bonding well to many substrate materials and of reasonable match in thermal expansion coefficient.

Degenerate Regions are regions within a semiconductor substrate in which the conductivity is very large and in which carrier transport is essentially ohmic. These regions are generally produced by alloying near saturation concentrations of a doping impurity into a specific pattern. Because of this method of formation the paths are generally at the surface of the substrate. By control of the intrinsic techniques mentioned, resistivities of 10^{-3} to 10^{-4} ohm-cm can be obtained and in general may be considered to be feasible for internal conduction.

Connections from one Integrated Circuit to the outside world or to another substrate are treated in the Section on Mechanics as in conduction of heat away from dissipative elements within the Integrated Circuit.

Resistance

Resistance may be provided in Integrated Circuits in at least four ways. The three intrinsic ways are by bulk resistivity, by transverse conduction in thin diffused back-biased regions within a semiconductor substrate, or by an epitaxial layer of opposite impurity back-biased with respect to the parent substrate. The extrinisic way is by deposited thin-film resistors on top of the substrate. Each method has particular advantages for certain applications and a complete Integrated Circuit capability requires mastery and evaluation of each. Both methods may be sensitive to their ambients.

Anodized Tantalum, Titanium, or Aluminum films provide an attractive method for obtaining highly precise resistance values. The initial deposit of these metal films is relatively thick and is easily formed by the vapor plating and vacuum deposition techniques previously cited. After deposition, the metal film can be trimmed to a precise value by anodizing the outer surface of the metal film to its oxide which is an insulating semiconductor. Since the oxide thickness is a direct function of anodizing voltage, a high degree control of the remaining metal film thickness can be obtained. Resistance films in the 10 to 500 ohms per square range can be obtained with ±1/4 per cent reproducibility. Accurate layout of these resistance films is achieved by the application of photoresist techniques to define areas of anodization. The unanodized film can be utilized to form conduction elements.

Tin Oxide films have reached sheet resistance values of over 5000 ohms per square. With conventional resistance patterns, values of up to 1.0 megohm can be achieved. The films are produced by hydrolysis in a technique that has been brought to a high degree of development.

Indium Oxide films have been produced by a two-step metallizing-oxidizing process involving a vacuum deposition of pure indium in a low pressure pure O_2 atmosphere followed by a low temperature (below 200°C) thermal oxidation for several hours. This low temperature technique has application in instances where higher temperature resistance fabrication would damage other temperature sensitive thin-film functions.

Nichrome films can be evaporated directly on clean substrates by volatizing the alloy from a tungsten heater. These films show excellent adhesion when substrate surfaces are heated to 300°C. Nichrome resistance films can be reproducibly deposited and show relatively high stability on standing. They possess an average temperature coefficient of resistance of 6×10^{-5} ohms per ohm per degree centigrade over the temperature range -50 C to 150 C. However, these films have low resistivities which limit their application to 500 ohms per square. In addition, the surfaces of unencapsulated Nichrome films are prone to a certain amount of corrosion and oxidation which limit compatibility with other thin-film circuit element fabrication. Specific geometries of Nichrome resistance elements are usually achieved by the use of mechanical masks during evaporation. These films are quite amenable to forming ohmic contacts with other metal films.

Bulk Resistivity makes use of the gross geometry and parent material of the semiconductor substrate. It is easily controlled but is suitable only for relatively low values of resistance based upon maximum usable resistivity of ~500 ohm-cm. Since bulk resistivity of 500 ohm-cm is not compatible with present semiconductor base or collector technology, a more realistic upper limit is 100 ohm-cm with the most compatible value probably lying between 10 and 30 ohm-cm. Bulk resistivity has the additional handicap of a complicated temperature coefficient. This temperature dependence can occasionally be turned to advantage in circuit applications for temperature compensation of specific types of circuit.

Diffused Back-biased Regions cover the class of resistors formed by transverse conduction within a thin diffused back-biased layer of semiconductor. They cover a much wider range of values and have additional advantages in their possible range of temperature coefficients. By control of the diffusion profile it is possible to achieve positive, negative, or substantially zero temperature coefficient at room temperature. The difficulty in using diffused resistors in practical circuits lies in their great sensitivity to the back-biasing voltage including the self-generated component of back bias caused by the voltage drop in the resistor. Where very high resistance of non-critical value is required, diffused resistors may have great value. Their other attractive application is in circuits needing an electrically alterable resistance.

Epitaxially Grown Resistive Layers can form resistive regions with useful characteristics. The method is similar to the diffused back-biased resistance described above but the epitaxial process promises better control of the junction characteristics. The epitaxial resistor requires a compatible masking technique during layer growth if mesa techniques and wet chemistry are to be avoided.

Capacitance

A number of methods have been developed for providing Integrated Circuit capacitance. Capacitance may be provided intrinsically by reverse-biased semiconductor junctions, by self-biased junctions or extrinsically by deposited thin-film capacitors using gold or some other conducting film as a counterelectrode.

Anodized Tantalum, Titanium, Aluminum, or Niobium can be used to form the lower conducting layer and dielectric of deposited thin-film capacitors. After the desired thickness of dielectric has been formed, a counterelectrode of some conducting material is deposited to complete the capacitor. The films are generally deposited by evaporation or sputtering. Subsequent anodization can be controlled to a high degree and pinholes cleared before deposition of gold as the counterelectrode. Ratings of 5.0 volt microfarads per square centimeter at 50% of breakdown voltage are now state of the art.

Titanium, Aluminum, and Niobium, can also be anodized with useful characteristics. Aluminum oxide has a dielectric constant less than 25% that of tantalum but has almost twice the working voltage for the same forming voltage. Unless the counterelectrode is of the same material as the anodized material, this type of capacitor is polar.

Deposited Metal Oxide Glasses can be sandwiched between deposited conductors to form an alternative thin-film capacitor. Here the dielectric constant is much lower than either of the anodized dielectrics mentioned above but the use of low melting silicate type glasses avoids some of the problems of the anodized capacitor process. Deposited metal oxide glass dielectric capacitors can be fabricated at lower temperatures than are required in the deposition of tantalum.

Deposited Ferroelectrics offer attractive possibilities as dielectrics for thin-film capacitors. Chief among the attractions is a dielectric constant for barium titanate three orders of magnitude greater than that for the silicate type glasses. Such a dielectric might also have important contributions stemming from its nonlinearity and polarizable nature. The chief disadvantages appear to be a limited operating range of temperatures and instability or deterioration of electrical properties. Work with deposited ferroelectrics is in the early exploratory stage.

Diffused Back-biased Junction capacitors depend upon the depletion layer as dielectric and, for step junctions at low voltages, can provide on the order of 1 volt-microfarad/sq.cm. However, since the width of the depletion layer varies with the magnitude of the reverse bias, such capacitors are electrically alterable with the magnitude of their effective capacitance depending on their bias. This voltage dependence can be an advantage or disadvantage depending on the use to which the capacitor must be put.

Reverse-biased junctions act as capacitors whose value depends not only on the junction area but also on the width of the depletion region at the junction. This width depends upon the impurity concentration gradient on each side of the junction and upon the reverse voltage applied across the junction. Since the normal diffusion process produces a graded junction, capacitors formed by diffused junctions tend to have lower values of capacitance at the same bias than either the alloyed or the epitaxially grown junctions discussed below.

If current flows parallel to the junction outside the depletion region, the bias on the junction will not be everywhere the same. In particular, if current flows through the bulk resistance of the substrate beneath a shallow diffused but otherwise unbiased junction, the floating junction will conduct to equalize potential at one end and will thus back-bias the rest of the junction. The result can be useful in circuits where a small capacitance is needed to bridge a load or coupling resistor. Note that this configuration is really a passive network of distributed resistance and capacitance.

Alloyed Back-biased Junction capacitors tend to have a much steeper impurity gradient across the junction on one side while maintaining practically the impurity concentration of the parent substrate on the other. The approximation to a step junction is much better than in the diffused case and much higher values of capacitance at low reverse bias are possible. Because of the relatively low impurity concentration of the substrate, the total depletion layer is very nearly as wide at large values of reverse bias as in the diffused-junction case.

Epitaxial Back-biased Junction capacitors hold promise of the ultimate attainable in reverse-biased junction capacitors. In addition to being more amenable to control than the alloy process, they are able more closely to approximate the step junction from high impurity concentration of one type to high impurity concentration of the other. This means that they should be capable of having the maximum capacitance, at low values of reverse bias, of any junction capacitor. It also means that their voltage dependence should be more predictable.

Thermal Oxidation on the surface of a semiconducting substrate can provide the dielectric for a hybrid type of capacitor in which one plate is the semiconductor substrate and the other plate is a thin metallic film. Such hybrids can have occasional applications as special networks.

Where precise and voltage-stable capacitors of less than a microfarad are needed, deposited thin films of tantalum, titanium, aluminum, or niobium can be anodized with good control. High-voltage or nonpolar capacitors with small values of capacitance can be fabricated reliably using silicate glass type dielectrics sandwiched between metal film electrodes. Voltage variable capacitors or non-critical coupling capacitors can be formed in semiconducting substrates by back-biased junctions.

Inductance

From an Integrated Circuit point of view, inductance is without doubt the most difficult to obtain of all the basic circuit parameters. The major difficulty is the requirement of a volume for the storage of magnetic flux which does not lend itself readily to Integrated Circuitry. Furthermore, the coupling of energy to the volume poses the requirement of a coil that is not easily achieved by deposited film techniques. Both a square-loop and a linear response are required for a general systems capability.

Deposited Nickel-Iron Films are among the extrinsic means available for storing energy resulting from the flow of current. Deposited nickel-iron films of 82%-18% composition and 1000Å to 4000Å thickness have been used for storing energy for logical matrices and as small-valued low-frequency inductors. A requirement is the presence of a magnetic field during the deposition process to orient the magnetic anisotropy. Unfortunately this limits the magnetic film configuration to simple forms. If the driving magnetic field is applied in the direction of the "easy" magnetization of the domains, a square-loop B-H curve results which is useful for memory and for magnetic logic applications. When the magnetic field intensity is applied normally to the direction of "easy" magnetization, a more linear B-H curve results which is useful for linear systems and impedance transformation.

Deposited Ferrites have possibilities as a second extrinsic technique for obtaining inductance. Glass, which is a mixture of metal oxides, is currently being deposited by the pyrolytic decomposition of suitable metallic-organic esters. The reaction temperature required to form the ferrite material from mixed oxides is in the order of 300°C. Since ferrites do not have the same magnetic anisotropy as thin nickel-iron films, no magnetic field is needed upon deposition and the form factor is not limited as it is with metallic films. Magnesium-manganese ferrite material provides a square-loop B-H curve and is, therefore, suitable for magnetic logic elements. Manganese-zinc ferrite provides a high μQ linear material usable to about 500 kc. Nickel-zinc ferrite provides a high μQ linear material suitable for the frequency range 0.5 to 100 mc. By proper masking methods it is possible to form thin-film solenoids which surround such deposited ferrite materials and provide a means for coupling energy into and out of the material. Finally ferrite materials possess variable permeability which is a function of the applied dc magnetic field, and this can provide a control element for ac magnetic flux. Such variable inductors can serve as electronic tuning elements or other control elements.

Air Core Geometries are suitable for r.f. coils and other high-frequency small-valued inductors. It is possible to deposit "air core" pancake-type windings of thin-film conductors. When associated with thin-film insulators of low dielectric constant the pancake-type winding can be formed in multilayers to increase the total inductance of the element.

Inversion of Capacitance by Active Element networks is among the intrinsic techniques for obtaining inductance. Field effect semiconductor devices can provide an impedance inversion function. By this means a low-Q capacitor can be made to appear as a high-Q inductor for circuit applications where resonance is not required.

Ferrite Substrates provide the possibility of using a single or multi-aperature ferrite material both as an inductive core and as a substrate for other Integrated Circuit elements. The coupling to the ferrite can be achieved by thin-film conductors. These elements, because of the dependence of their permeability on the applied field, can also serve as a means for controlling magnetic flux.

The Inductance Diode is another intrinsic approach to the problem of providing inductance for Integrated Circuits. Although this device is still in the early experimental stages a number of workers in the field have succeeded in measuring inductive behavior of germanium diodes. The maximum effect is found with alloyed or near-step junctions in which the injection efficiency is close to unity. Chief difficulty in measuring or applying the observed inductive behavior is the instability of the inductive effect, both with current through the device and with temperature.

The problems surrounding the incorporation of inductance in Integrated Circuitry are severe and first efforts at Integrated Circuit design will probably try to accomplish the desired equipment functions without the use of inductance.

Special Networks

A few basic circuit parameters are of a hybrid nature and do not fall logically into any of the pure parameters discussed above.

Deposited Distributed R-C networks are among the extrinsic examples of functions that cannot easily be duplicated by lumped constant circuits. A simple example of this is an anodized tantalum resistor upon which has been deposited a counterelectrode of gold or some other metal. Each part of the resistor has a capacitive relationship through the anodized dielectric to a unipotential conducting plane. The result is a series resistance with distributed capacitance acting all along its length.

Deposited L-C networks are in the same class of difficulty as ordinary thin-film lumped inductance. About the only simple example is the pancake air core coil deposited as the counterelectrode of a deposited and anodized tantalum capacitor.

Transformer-like thin-film configurations have been proposed but both distributed L-C and thin-film passive impedance transformation for Integrated Circuitry are in the early experimental state.

Diffused Back-biased R-C networks are similar to the thin-film extrinsic kind except that they are voltage sensitive as to their capacitance and sometimes even their resistance. If the current-carrying diffused layer is thin enough, or if the back-biased junction is between the parent substrate and a thin current carrying epitaxial layer, variation of the reverse-biasing voltage will affect both the series resistance of the thin layer and the shunt capacitance to the substrate. Such a configuration can form a tuning unit in a phase-shift oscillator or feedback amplifier.

Bulk Resonance effects such as the piezoelectric resonance of specially cut quartz crystals are classed as intrinsic special networks even though they normally occur in other then semiconducting substrates. In the case cited, the mechanical motion of the crystal would probably be disastrously damped by using it as the substrate for other Integrated Circuits, but it is conceivable that thin-film circuits could be laid out entirely along nodal lines.

Active Impedance Transform Networks cover a wide and largely undeveloped field. Their scope extends from the impedance transformation characteristics of ordinary non-integrated devices such as transistors and field effect devices, to vague and esoteric proposals for active delay networks and hybrids of active and passive phenomena.

By definition, the functions included in this category do not in general exist in a one-to-one correspondence outside the solid state. For this reason they are perhaps closer than some of the other basic circuit functions to being Integrated Circuits themselves. The members of this category are expected to increase as new integrated phenomena are developed.

2.1 Active Elements and Substrates

The active semiconductor elements useful in Integrated Circuitry include the logic diode, the breakdown or zener diodes, various multilayer diodes, the mesa transistor structure, the flat or oxide-masked mesa, the unipolar transistors, various other field effect devices, and compound transistor-like elements. The list of non-semiconductor active elements includes ferromagnetic and ferroelectric elements and many others.

Vapor Deposition of Semiconductor Active Elements on a passive substrate is undoubtedly the most important extrinsic technique in this category.

To date, no one has reported on the deposition of large-area nondegenerate single-crystal thin films of semiconducting material on an insulating substrate. When this technique is fully developed it will be possible not only to build complex active functions but to build them on either active or passive substrates as the situation dictates. One of the most important consequences following on the eventual achievement of the thin-film deposition of active semiconductor elements will be the removal of the topological-electrical limit to the number of active elements that can be built into one semiconductor substrate. Another will be a decrease in the minimum capacitive coupling attainable in an Integrated Circuit.

Vapor Deposition of Nonsemiconductor Active Elements on a passive substrate can make useful many previously overlooked solid state effects. Both ferroelectric and ferromagnetic materials are unique in that certain of their basic properties, such as dielectric constant or permeability, can be readily changed by the application to the material of an electric or magnetic field of the proper magnitude. This property has been utilized in microcircuits for the purpose of circuit tuning. When the problems of the deposition of coherent films of these materials are overcome, an entirely new generation of active circuit elements will be possible, utilizing the nonlinear properties of ferroelectrics and ferromagnetics to actively tune circuits, act as band-pass filters, and so forth.

Applique of Active Elements has been the universally used extrinsic expedient since thin-film semiconductor active elements do not yet exist on insulating substrates. The active elements are attached to passive substrates to make a hybrid structure. They may either be conventional or special microminiature elements in cans with leads brought through the passive substrate and soldered, or they may be unencapsulated or self-encapsulated devices mechanically and electrically affixed on the passive two-dimensional substrate.

Conventional Active Element Design can in general be transferred directly to Integrated Circuit application. Intrinsic techniques for creating active elements within semiconductor substrates include doped or rate-grown junctions in crystals as pulled from the melt, alloyed junctions, diffused junctions, vapor-grown junctions as epitaxially oriented overgrowth and hybrid junctions where one side of the junction is the result of one process and the other of another.

Epitaxial Techniques of growing silicon layers on silicon parent stock are among the most promising tools in the area of three-dimensional Integrated Circuits. Vapor phase epitaxially oriented overgrowth was first developed in Europe and has since been intensively studied and applied by American materials and components manufacturers.

The epitaxial process permits a number of desirable improvements over conventional active element technology. The previous requirement with active semiconductor substrates that the active elements be fabricated within the substrate becomes within or on the substrate. The substrate can now be a lower resistivity than could be tolerated when it had also to form the collector junction of our active elements. Large-area step junctions are now much more nearly a reality and the only diffusion effects are those occurring at the junction while the epitaxial layer or layers are being grown.

There are many facets of the process that need exploration and development but the prospective rewards are great - both from the standpoint of new and hitherto unfeasible active and passive elements and from the standpoint of logical extension to automated, low cost, and highly flexible Integrated Circuit assembly processes.

Form Controlled Growth. In the extension of their studies of semiconductor crystal growth a number of companies are developing methods of ribbon crystal growth. The basis of all this work is the ability to control the shape and structure of a growing semiconductor crystal as it is drawn from the melt. In some cases the ribbon is a dendrite pulled from a supercooled melt, in other cases different means of obtaining ribbon shape are used.

Ferrite Substrates. The utilization of ferrite substrates offers a unique method of introducing an inductive element into the circuit. The requirements of such a substrate are those which must be met by any substrate, such as surface smoothness, compatibility of thermal expansion and thermal conductivity with other active elements and physical strength. The use of a magnetic substrate imposes additional requirements. Magnetostrictive effects might present the problem of maintaining a bond between the substrate and deposited films, as well as changing the electrical properties of the films through their electrostrictive properties.

In order to take full advantage of a magnetic substrate, the magnetic properties of the material must be matched to the particular requirements. For inductive purposes a low loss, high permeability material would be required, suitable for use at the frequency of interest. The temperature coefficient of permeability might be important where large fluctuations of the ambient temperature are anticipated.

The geometry of the substrate would depend upon the method used to couple the magnetic field into the circuit. One method consists of laying down the coil as a thin film on the surface. Another consists of providing holes in the substrate through which, or around which, wires can be wound.

Encapsulation

The encapsulation problem is a critical one. Achievement of desired performance, control, and reliability of semiconductor active elements, and to a lesser extent passive components, is dependent upon reliable and well-founded solutions to the protection of surfaces from the migration of contaminants in the ambient. These include moisture, radiation, and many other forms of contamination.

Hermetic Seal has been the most widely used and until recently the only satisfactory means of protecting semiconductor devices from contamination. In the initial work with Integrated Circuitry the hermetic seal will undoubtedly continue to be used. A number of companies have made great strides, however, in developing other means of encapsulation and the hermetic seal should be considered only as a temporary expedient.

Low Melting Inorganic Glasses have shown remarkable success in improving device performance and reliability. The glass may act as an ion getter in cleaning up surface contaminants and it protects against 100% humidity for over 10,000 hours. Protective films can be deposited from the vapor phase over large surface areas. Their application is limited, however, by the facts that they may be soft at room temperature, that sulphur in the glass reacts detrimentally with certain thin-film materials, and that thermal expansion in mating with lead materials may cause difficulties.

Inorganic Film Pyrolysis offers a number of advantages: it is free from pinholes, does not affect the device, has high dielectric strength, has high chemical and physical durability, has low volume permeability to moisture, and requires only low device temperature during encapsulation. Although the work with pryolitic glass is not widespread, there are indications that this approach will have wide applicability to the physical and chemical protection of Integrated Circuit Functions of the future.

Anodization has already been discussed under Insulation, Resistance, and Capacitance. It uses the parent metal in the oxide formation.

Accelerated Thermal Oxidation is a moderate temperature reaction carried out at atmospheric pressure which can form a completely protective glass type layer by conversion of in situ material. A characteristic of this process is that it applies only to material capable of being converted to glass, and hence is most applicable to silicon surfaces. In the self-encapsulation of a silicon device, therefore, the process leaves substantially untouched the leads or contacts. This feature has advantages for straight device fabrication but is of restricted applicability where thin films and semiconductors are combined in Integrated Circuits.

Surface Stabilization is an attractive possibility involving the deliberate addition of something to the surface of an active element which ties up all the possible loose ends and leaves the surface indifferent to subsequent contamination.

Mechanics

The mechanics of Integrated Circuitry are as much a technique as the realization of any circuit element. They include form factor, masking, thermal conduction, access, and interconnection of the Integrated Circuits with each other and with the outside world.

The Form Factor includes the size and shape of the substrate and must take into account such things as heat removal, replaceability, accessability for test, position of leads, and so forth. The initial form factor must allow for hermetic seal of each Integrated Circuit. A second step will probably be individual surface passivation or self-encapsulation with a number of related circuits in one can for protection. Ultimately, surface passivation will probably also constitute physical protection and the individual circuits will be entirely self-encapsulated.

Masking is another area falling within the field of Mechanics. Masking is chemical insulation during the fabrication process. Chemical insulation as used here refers to means of restricting the action of chemical etches and to masking portions of the substrate during evaporation, alloying, and/or diffusion. Metallic masks are a standard part of the art for evaporation as are the photoresist techniques and wax coating for etch protection. Recently added as standard art are self-oxidation and vapor deposition of glass which give promise of simplifying mask technology by an order of magnitude and reducing complicated micro layout to a photographic reduction of standard art work.

The Interconnection of two or more Integrated Circuits is one of the most important parts of the program. It may well be the most important consideration facing all of Integrated Circuitry at this time. Interconnection covers not only the communication of the Integrated Circuitry with associated equipment but also the communication and supply of power between substrates. As such it includes some of the most important and perplexing problems in the field. If size, weight, and reliability must remain tied to our present method of plugs and interconnecting wiring, much of the impetus of Integrated Circuits is lost.

Present efforts make use of ohmic connection,in some cases in the form of strips of conductor sandwiched between isolating ground planes. Such low impedance strip lines have a number of advantages from the circuit standpoint but the difficulties of removal and replacement of a particular circuit are formidable. Ultimately, other means of coupling (in addition to ohmic connection) will be used. Magnetic, capacitive, photon, and ultrasonic coupling are possibilities as means of conveying information and power.

Designability

The development of an Integrated Circuitry depends not only on being able to fabricate and interconnect the basic circuit parameters discussed above but also on knowing what techniques are compatible with which, what range of parameter values are realizable, what reliability individual circuit functions may be expected to have and what control is possible in the constituent processes. It depends on the design of processes that fit into automated manufacturing methods. It depends on building into the manufacturing methods, from initial conception onwards, sufficient flexibility that the output of the product line can be changed easily and quickly to yield a different Integrated Circuit.

Finally, the widespread development of Integrated Circuitry depends on being able to accomplish all the necessary functions and operations economically. In the last analysis, it is cost that will determine the acceptance and use of Integrated Circuitry. If a particular fabrication process cannot be made economically competitive, it is the wrong process.

Conclusion

Microsystem electronics has struggled through a variety of stages on the way to maturity. Undoubtedly it has a long way yet to go. But the need, significance, and accomplishments to date are such that it can no longer be ignored by the computer industry. For a while it will be used mainly by those for whom nothing else will do, primarily because of size and weight. After reliability is proven there will probably be another period when the main use will be in military applications. Developments during this period will determine whether Integrated Circuitry spreads throughout our electronics industry or remains with a few specialized applications. The determining factor will be cost. If the ultimate cost can be brought down to that of conventional computer circuitry, Integrated Circuitry will take over large segments of the electronic computer industry.

Such an acceptance of Integrated Circuitry will force another integration. During the proving period computer manufacturers will probably be willing to design their equipment around available integrated circuits, but the time will come when the computer design engineer will demand greater freedom of choice in selecting the particular integrated circuits from which he builds his machine. When this happens, and it surely will happen, our industry will go through some interesting gyrations; computer equipment companies will be pushed into the semiconductor and solid state materials and components business, and the semiconductor products manufacturer will be forced into the systems engineering and equipment fabrication business. The end point will be remarkably similar for the two, even though there will always be differences in accent and degree. When dynamic equilibrium is reached, not only the electronic circuitry but the people who create and use the new circuitry will be integrated to an extent unknown today.

There will always be specialists, as there always have been, but the new specialists will have a broader base which will frequently extend into more than a single scientific or technological discipline. The new specialist will understand and speak the languages of the associated disciplines and will contribute his point of view in matters that previously were considered the exclusive concern of others. The chemist and the metallurgist will take their rightful places beside the solid state physicist, the circuit designer, the systems engineer, and the statistician.

The successful integration of computer circuitry will depend upon demonstrating adequate control of designable, compatible, and above all reliable techniques.

Acknowledgment

In a survey paper such as this, the work and techniques described are the result of effort by many people in many companies. In particular I would like to mention the help and cooperation of my colleague, James R. Black of Motorola's Solid State Electronics Department, in supplying information on thin-film integrated circuit techniques. Finally, to all technologists whose contributions are included in these pages, I wish to express appreciation for sharing their discoveries and experience. The rapid growth of microsystem electronics depends upon just such sharing and cooperation.

TESTING OF MICROLOGIC ELEMENTS

Robert H. Norman and Richard C. Anderson
Fairchild Semiconductor Corporation
Palo Alto, California

Summary

The acceptance tests of integrated logic-function elements require a departure from the notion that the parameters of the individual components of a circuit must be known. A test program based on external characteristics only is made necessary by the complexity of the circuit; it is contended, in addition, that such a test has more valid significance than does measurement of constituent parameters. It is more significant, for example, when testing a compatible set of digital functional blocks, to make a measurement which gives a direct estimate of fan-out than it is to measure the gains, leakage currents, and resistor values in the output stages.

This paper will discuss the validity of this concept and describe a program of tests based on the concept.

Introduction

In the production testing of integrated logic-function circuits, we face two familiar problems: first, the test must insure that the device operates properly over a wide range of environments, yet it is desirable that the test be performed at room temperature. Second, the test must not only reject those which are initially inadequate, but it must reject those which are predictable future failures. In addition to these usual requirements, in testing functional blocks a third problem soon becomes evident: the integrated logic circuit contains some active and passive constituents whose individual parameters cannot be measured, due to the circuit in which they are incorporated. Further, the device contains several internal circuit nodes which are not accessible for measurements once the device is packaged and complete. In short, in each circuit some device parameters would be unmeasurable if we could get to them . . . and we can't.

This, then, is the new problem posed by the functional-block circuit: the complexity of the device dictates that no meaningful reliability tests and evaluation can be performed unless they be tests of external characteristics. We can no longer be concerned, for test and evaluation purposes, with the parameters of every active and passive constituent of the circuit--with the gains of the transistors, for example--or even, for that matter, with the distributions of those parameters. We must be able to test a completed integrated circuit by examining only its external characteristics in such a way that we can accurately determine that its present operation is satisfactory and, as important, that we can expect continued operation over wide environmental conditions. If this cannot be done, there is little point in continuing with an integrated circuit program.

We will support the assertion that this can indeed be done by discussing a test program which is based upon the supposition that the assertion is correct. This will not, of course, constitute proof; it will give, we hope, an intuitive confidence that the probability of success is high. A theoretical proof of the adequacy of an "external characteristics only" test would be both difficult and unacceptable. The only ultimate verification of the adequacy of such a test procedure will be its success in operation after millions of device-hours.

Micrologic

The remarks above are general, readily-apparent comments concerning the entire genus of integrated function-block devices; the tests and methods now to be discussed pertain to a particular species, the micrologic element.

Micrologic is a compatible and sufficient set of integrated semiconductor logic-function elements. Each micrologic element consists of from one to five DCTL NOR gates; the circuit components--resistors and planar transistors--are diffused into a single slab of silicon, and metal intraconnections are deposited on top of the slab. The device is then packaged in an eight-lead TO-5 or TO-18 package. The logic and schematic diagrams of a typical element, the flip-flop, are shown in Figure 1.

Effects of Parameter Distribution and Drift

Before the micrologic development program was undertaken, some assurance was required that the problem under discussion could be solved. Since the circuit configuration chosen can tolerate a wide variation of parameters, and since these parameters might not initially be measurable, assurance was required that those parameters far from the expected value would not drift with time, causing subsequent failures. Intuitively, the inherent stability and reliability resulting from the planar process gave this assurance.

To investigate these questions, flip-flops were made up of conventional components, using the circuit configuration proposed for the future integrated "F" element; the transistors used were

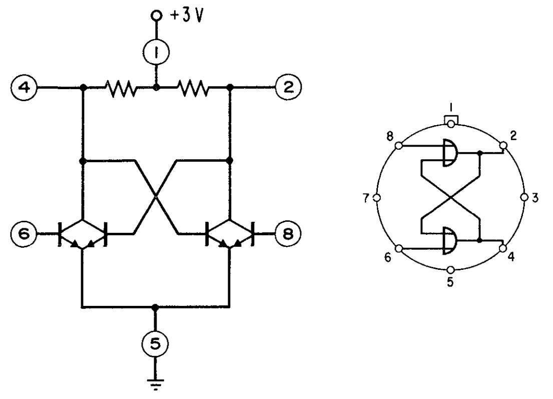


Figure 1. Schematic and Logic Diagrams of the Micrologic "F" Element.

reject devices. Approximately half were V_{EBO} and I_{CBO} rejects and half were random rejects. The V_{EBO} rejects gave a 20% yield of good flip-flops, the I_{CBO} and random rejects gave an 80% yield, indicating that the DCTL circuit configuration is relatively insensitive to wide parameter variations. One hundred fifty of these circuits were then placed on 150°C storage test, together with 25 similar circuits made up of good transistors.

After 7000 hours at 150°C ten of these flip-flops have failed to operate under load. Seven of these failures were caused by leads and three by EB shorts. There were no other modes of failure. Environmental tests indicate that these failure modes will not be a problem with the deposited-metal connection method developed for micrologic elements. The results to date in this life test give us, therefore, a reasonable degree of assurance that transistor parameters which are outside specifications will not drift with time sufficiently to cause operational failure in the micrologic DCTL circuit.

This first-step program of building simulated micrologic elements thus indicated that, indeed, individual-constituent parameters are not the primary criteria for estimates of operability, and that circuits which are initially good, although they may have "poor" constituent parameters, will stay good.

Micrologic Test Program

The test program for micrologic elements is illustrated by Figure 2, which shows the flow of completed elements through the three types of measurement phases:

1. The LOGIC SORT classifies elements as to type and rejects catastrophic failures. It is a go, no-go test with no data recording.

2. The PRODUCTION TEST is the phase that incorporates the test principle we have been discussing. It is a sequence of DC measurements of the overall input-output characteristics of the element, with no attempt made to measure internal parameters. It is designed to determine the present and expected operability of the element. A go, no-go indication is given for each element, and all measured data are recorded.

3. The LAB TEST is a measurement of all of the internal device parameters, such as β, rb, R_c, that are accessible. These parameters are measured and recorded over the specified temperature range.

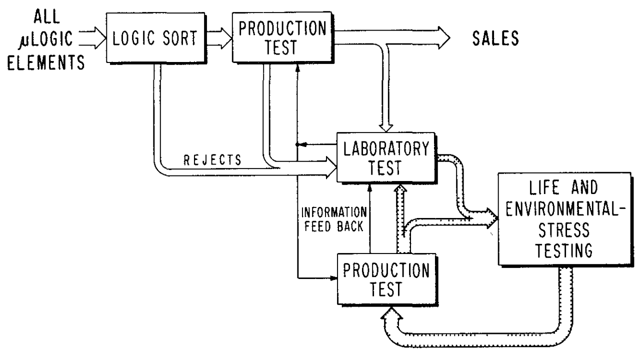


Figure 2. Flow Diagram of the Micrologic Test Program.

It is relatively easy to set up a production test that evaluates present element operability; to devise a test that maximizes future operability over a long time span, and under wide temperature variations is a more difficult matter. Only with time can the accuracy and significance of such a test be verified. For these reasons the life-test production-measurement loop has been set up, as shown in gray in Figure 2. Sample quantities of the elements which are accepted by the Production Tester are fed into this loop; the elements of the sample are run through the lab test and all parameters are recorded. These elements are then placed on operating and storage life tests. At given intervals, they are again run through a production test, samples are again lab-tested, and the group is returned to the life test.

It is this loop that generates the feedback information shown in the figure. If the production test indicates a drift with time, the sample sent through the lab test is increased in size, and the parameter causing the drift is studied. If the lab tests, made over the temperature range, indicate that the room-temperature production test requires modification, then the production-test parameters, or criteria, or even test methods are altered.

The methods, parameters, and criteria of the production test, then, are initially set up according to our best first guess. The recirculating life-test, lab-test loop generates the feedback information that tightens or relaxes the production test.

As the rate of predictable failures in the loop decreases, our confidence in the validity of the production test measurements increases.

Let us now consider in greater detail each of the phases of the test program.

Logic Sort. The Logic Sorter, shown in Figure 3, uses a four-stage counter to generate

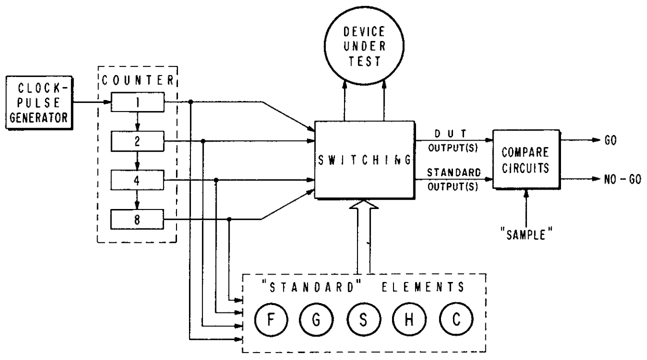


Figure 3. Block Diagram of the Logic Sorter.

binary inputs which are applied to six "standard" elements and to the switching network. As the switch steps through the test sequence, the device under test is tested first as an F, then as a G, then as an S element, the correct inputs being supplied by the switch, and the outputs being logically compared to the outputs of the corresponding "standard" element.

The need for a logic sort test arises from the fact that micrologic elements may come from the assembly process with no classification as to type. Since the only differences among the processes for the various elements are in the masks used, no attempt at classification need be made from the time the wafer is diced. When an element of unknown type is logic-sorted, the tester steps through its sequence, and, when a correct comparison with a "standard" is found, the element is sorted as that type. If the sequence ends without a comparison, the element is rejected. The sorter rejects any device that does not give a logically correct output when loaded by a fan-out of two at room temperature. Thus it acts both as a convenient sorter and as a measure to prevent elements with major flaws--such as short-circuits--from reaching the Production Tester, where they might cause overloading.

Production Test. As mentioned above, it is the Production Tester that embodies the principle of rejecting incipient failures from a series of measurements of external characteristics. The tester automatically steps through a different sequence of tests for each element, setting up worst-combinations of applied conditions, recording the resultant D.C. measurements, and providing go, no-go indication and recording.

The establishment of a production test is essentially the definition of the criteria of what constitutes a satisfactory device. That is, the setting of acceptance limits implies that these limits are the valid and significant criteria by which one can evaluate the usability as well as the reliability of the device. For these reasons, we need first to discuss the factors that govern our selection of the general criteria of usefulness and reliability and then to discuss the DCTL circuit characteristics and limitations that determine the specific criteria; then we can return to discuss the production test conditions and acceptance limits.

General Criteria

The general criterion of acceptance of a "good" micrologic element is based on the following definition: if any given system operates correctly, utilizing a given element, then that element is considered to be good. This definition is, of course, a very general one, usable only under certain conditions, namely that "correct operation" of the system can be clearly defined and that the set of all circuit environments of an element in any given system is a limited one, permitting description of all possible circuit configurations or of the most stringent configurations.

The nature of the micrologic family is such that these conditions are met. "Correct operation" of the arbitrary system can be definitely specified because it is a binary system; "correct operation" is, therefore, synonymous with error-free operation.

The set of all possible circuit environments is known because micrologic elements alone are used in any micrologic system, and their use in the system is constrained by the logic design rules. These design rules specify that no more than six transistor collectors may be connected to a given node (i.e., fan-in), and no more than five micrologic loads may be driven by a given node (i.e., fan-out). The most severe input-output conditions for any element can, therefore, be readily derived. In fact, by analyzing several micrologic digital systems, one can determine the frequency-of-occurance distributions of the many combinations of fan-in and fan-out. A study is now underway to determine these distributions in a typical small general purpose computer and in several logic subassemblies.

In summary, a micrologic system consists solely of micrologic elements, and the set of all possible combinations of these elements is limited by the design rules, permitting accurate description of the limits of the circuit environment that will be encountered by any given element. These factors permit the use, for micrologic, of this general criterion of acceptance: the element is good if any arbitrary system utilizing it operates correctly.

Direct-Coupled Transistor Logic

The basic circuit, and the basic logic block, used in micrologic is the DCTL NOR gate, shown schematically and in logic symbol in Figure 4.

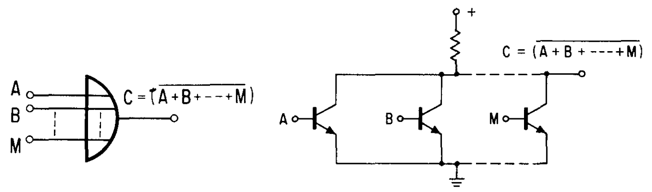


Figure 4. The DCTL NOR Circuit and Logic Symbol.

The circuit is a simple one, consisting of a transistor for each input and a single collector resistor. The output of the gate is a "one" (a positive potential) if none of the inputs is a one.

DCTL was selected for use in micrologic because of its simplicity, low power consumption, high speed, and insensitivity to wide variation of certain parameters. One limitation of DCTL is the requirement for a transistor type which has a narrow distribution of base current for a given value of "on" base voltage. This requirement is necessitated by the "current hogging" characteristic discussed below. One phase of the micrologic program was the development of such a suitable DCTL transistor.

Current Hogging. This euphonious title applies to the situation shown in Figure 5.

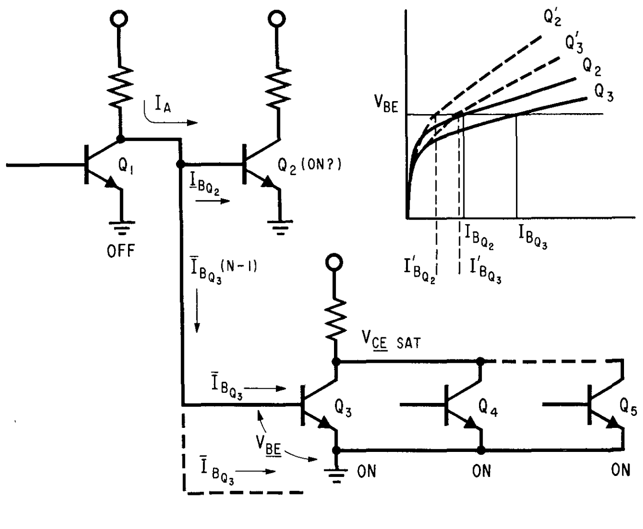


Figure 5. The "Current-Hogging" Condition.

Here transistor Q_1 fans out to N loads, one of which is a simple inverter, and (N-1) of which are multiple-input gates, all of whose transistors are in saturation. The simple inverter has the V_{BE} / I_B characteristic marked Q_2 on the curves, but the other (N-1) loads have the input characteristics of Q_3, because their collectors are clamped by the parallel transistors. For a given node voltage V_{BE}, therefore, the inverter base will draw current $I_{B(Q_2)}$, while each of the other (N-1) bases draws a far larger current, $I_{B(Q_3)}$. There is then a good possibility that the heavy current drawn by the (N-1) loads will not permit the voltage V_{BE} to rise high enough to fully turn on Q_2.

If, however, the base input characteristics are closely uniform and if the base input resistance is increased moderately, then the disparity in input currents is greatly reduced, as shown by the dotted curves. When these input characteristics are incorporated in the DCTL transistor, the current-hogging problem is minimized, and the advantages of DCTL are made available.

Specification of Threshold Levels. In the design of a logic circuit some specification must be made of the signal levels representing 1's and 0's. In DCTL circuits, a one is represented by a positive potential, the level of which is determined largely by the bases being driven. A zero is represented by a near-ground potential, the value determined by the saturation voltage of the output transistors and the number of such transistors in parallel.

The degree of saturation in the "on" condition and the permitted amount of collector current in the "off" condition are, within limitations, up to the designer, and there are many combinations of "one" and "zero" voltages that might appear to be usable. Just what are the limitations?

In DCTL, we are concerned, of course, with a distributed system, consisting entirely of closely similar gates. When we determine the limiting values that the signal levels may take, the solution must hold over every logic path in any system.

The problem to be solved can be considered in this manner: assuming an infinite cascade of DCTL logic stages, what is the lowest "turn on" voltage and the highest "turn off" voltage that may appear at any base, under the condition that all following stages in the cascade are alternately on and off. Considering Figure 6, what is the highest voltage that we may apply to the base of Q_0 and still have the condition that Q_0

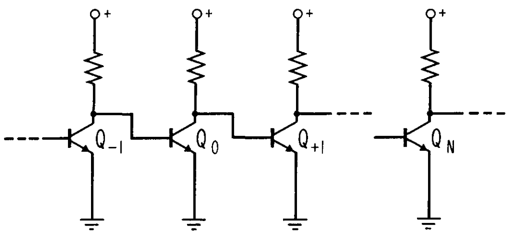


Figure 6. Cascaded DCTL Inverters.

and all even-numbered transistors are cut-off, and that Q_1 and all odd-numbered transistors are on? Similarly, what is the lowest base voltage applied to Q_0 that insures that Q_0 and all even-numbered devices are "on" sufficiently to hold the odd-numbered devices cut-off?

This problem can be graphically analyzed by considering the circuit of Figure 7 together with the curves of Figure 8. The curves indicate V_{CE} as a function of V_{BE} for a representative group of DCTL transistors with collector resistors; the upper and lower limits are the 3σ points of the distribution. The two-transistor feedback loop of Figure 7 simulates the "infinite" cascade; transistor A has a V_{CE} characteristic corresponding to the upper 3σ curve, and B is represented by the lower curve.

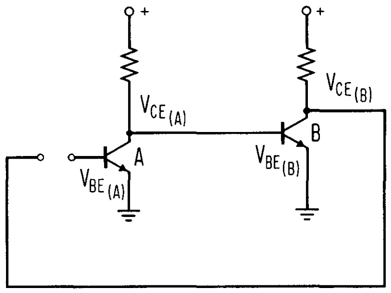


Figure 7. Two-transistor Feedback Loop.

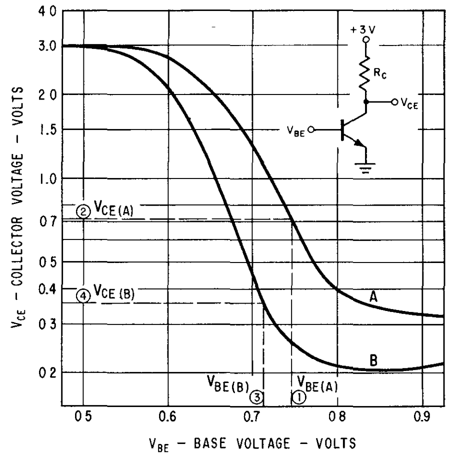


Figure 8. Curves Showing 3σ Limits of the Distribution of V_{CE} as a Function of V_{BE} for a Representative Group of DCTL Transistors with their Collector Resistors. The First Trial Solution for the Threshold Value of V_{BE} is also Shown.

Assume initially that A is cutoff, B is conducting and that the feedback loop is open; a trial turn-on voltage, $V_{BE(A)}$, applied to A will result in $V_{CE(A)}$ at the collector of A and at the base of B. For $V_{BE(B)} = V_{CE(A)}$, the output of B is seen to be $V_{CE(B)}$, which is lower than the initial assumed applied voltage. The applied voltage was not sufficient, then, to switch the stable states had the feedback loop been connected, and it would not have been sufficient to drive a long cascade.

In Figure 9 let us assume a higher turn-on

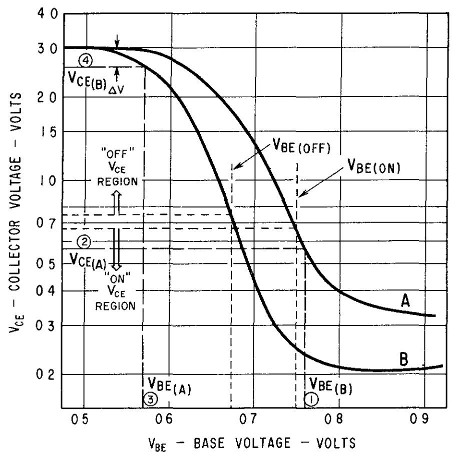


Figure 9. Curves of V_{CE} vs. V_{BE} with Second Trial Solution (Dashed Lines) and Threshold Operating Point (Dotted Lines).

voltage $V_{BE(A)}$ for the same circuit. Repeating the same steps (by proceeding from ① to ④), we find that the resultant feedback voltage is higher than the assumed turn-on voltage, so that--had the loop been connected--the voltage $V_{BE(A)}$ was more than sufficient to insure a reversal of stable states. This solution is indicated by dashed lines in Figure 9. By repetition of this graphic procedure for intermediate values of turn-on voltage, we can find the threshold value, $V_{BE(on)}$, which is just sufficient to turn on A enough that B is sufficiently cut off to cause its collector to rise to $V_{BE(on)}$. This threshold condition is shown by the dotted lines in the figure.

Since A has the poorest turn-on characteristic, we see that this is a worst-case condition within the 3 σ distribution and that all other transistors are turned on harder than A for the same minimum-required $V_{BE(on)}$.

Selection of this threshold value of turn-on voltage also implicitly determines the maximum voltage which may appear at the base of an off transistor. By selecting $V_{BE(on)}$ we have defined the "on" condition as one in which V_{CE} is $V_{CE(on)}$ or lower, as shown in the "on" V_{CE} region in Figure 9. Since this voltage is equal to the base voltage of the following stage, we see that the base voltage of an off transistor must be $V_{BE(off)}$ or lower.

These two values, then, $V_{BE(on)}$ and $V_{BE(off)}$, define the threshold levels for the on and off transistors in the feedback loop or in the cascade. These same values would be obtained if we assumed an initially-applied voltage to turn off B.

All those readers with experience in logic circuitry are now entitled to rise up in wrath, shouting, "What, no noise rejection?" and "What happened to fan-out!" together with mutterings of "What did I tell you about DCTL!" And this would be justified. The graphic solution above is a preliminary one; it is for the purpose of giving a basic understanding of the method of analysis of cascade of DCTL networks by consideration of the device parameter distributions. It does not consider loading effects or the necessity for noise rejection. These we will discuss next.

The Graphical Solution Considering Loading. Consider again the two-transistor feedback loop and the V_{CE} vs. V_{BE} curves of Figures 9 and 10. This time the output node of B has a current load. We again assume the trial turn-on voltage, $V_{BE(A)}$, which is higher than the threshold value. Following points ① through ④ in the figure, we see that the assumed applied voltage results in the appearance of $V_{CE(B)}$ at the collector of B.

Now these V_{CE} curves were plotted for an unloaded collector node; it is quite apparent that, in the circuit shown, the base A will not allow $V_{CE(B)}$ to rise to the value shown on the graph, but will clamp it to some lower value, $V_{BE(on)A}$. There is, however, useful information to be gained from the value of $V_{CE(B)}$; it indicates the collector current drawn by transistor B with $V_{BE(B)}$ applied to its base. This current, $I_{CEX} = \frac{\Delta V}{R_c}$. (See 4 of Figure 9)

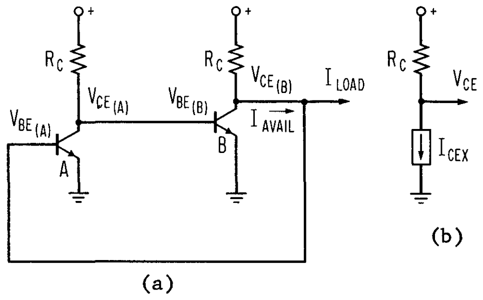


Figure 10. (a) Two-transistor Loop with Load.
(b) Equivalent Circuit for B.

To a first approximation, then, we can consider transistor B to be a current generator, drawing current I_{CEX} from its output node, as indicated in Figure 10b. In order that the assumed voltage $V_{BE(A)}$ be sustained while a load current is drawn from the B output node, the following is true:

$$I_{available} = \frac{V_{cc} - V_{BE(A)}}{R_c} - I_{CEX}, \qquad (1)$$

where $I_{available}$ is the current available from node B to drive the base A as well as other unspecified current loads at a voltage $V_{BE(A)}$. This current is easily calculated.

If this available load-driving current is plotted as a function of V_{BE}, we get a plot similar to the one of Figure 11, where the distribution of I_B as a function of V_{BE} is also shown. $I_{available}$ in this plot is the current available from node B at a given voltage V_{BE}, with the same voltage V_{BE} applied at the base of A. It is seen that as the operating V_{BE} point is reduced from a high value, the current-available gradually rises, since more current can be supplied through R_c to a lower voltage. As the $V_{BE(on)}$ operating point is approached, however, the "leakage" current drawn by the "off" transistor, B, becomes dominant, and $I_{available}$ falls extremely rapidly.

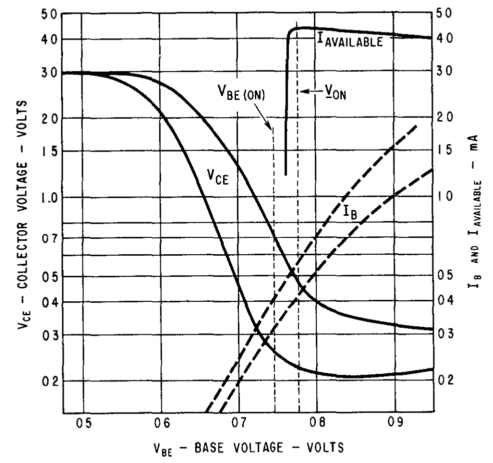


Figure 11. Collector Voltage, Base Current, and Current Available as Functions of Base Voltage for a Group of DCTL Transistors.

Knowing $I_{available}$ and I_{BE} as functions of V_{BE}, we can quickly determine fan-out as a function of V_{BE}. Clearly, for any operating point of V_{BE} the number of transistor bases that can be driven by a node is equal to the total current available divided by the current required by each base. Fan-out, N, can therefore be expressed by equation (2):[1]

$$N = \left. \frac{I_{available}}{\bar{I}_B} \right|_{V_{BE}} \qquad (2)$$

where $\bar{I}_B$ is the upper 3 σ value of base current at the given V_{BE}. Since this function has a maximum in the neighborhood of the maximum value of $I_{available}$, the specified value of the minimum

[1]A more accurate value for fan-out is obtained by using the upper 3σ limit of the distribution of $N(I_B)$, as in equation (3). Here N is an implicit function of $I_{available}$.

$$N(\bar{i}_B) + 3\sqrt{N}\,\sigma_{iB} = I_{available}, \qquad (3)$$

where N is equal to fan-out, $\bar{i}_B$ is the mean value of base current, and σ_{iB} is the standard deviation of the base current distribution. This gives a higher value for N; the use, therefore, of the simpler equation (2) results in a conservative estimate of fan-out.

acceptable "on" value of V_{BE} is established there. This value is defined as $\underline{V}_{on}$; it, in turn, specifies a maximum acceptable turn-off value of V_{BE}.

To recapitulate, $\underline{V}_{on}$, the minimum allowed base "on" voltage, is the voltage required to turn on all transistors sufficiently well that all collectors are $\bar{V}_{off}$ or lower; this voltage is, in turn, low enough that all transistors to which it is applied have V_{CE} equal to or greater than $\underline{V}_{on}$ while supplying current to drive the test-specified fan-out.

Note that, whereas $\underline{V}_{on}$ is the voltage required to turn on a "poor" inverter, the loads $\bar{I}_B$ in equation (2) are those required by transistors with clamped collectors. We have, therefore, based our calculations on the most stringent DCTL fan-out configuration.

In summary, we have graphically analyzed the distributions of V_{CE} and I_B as functions of V_{BE} in order to determine the available current and the fan-out as functions of V_{BE}. By establishing a minimum turn-on voltage and a maximum turn-off voltage which may be allowed to appear at a transistor base, we have insured that the calculated value of fan-out will be achieved at every node.

The methods of calculation of current available and fan-out discussed above are largely of academic interest. They are included here to give a better understanding of the tests we are about to discuss and to illustrate the evaluation procedures used during the development of the micrologic DCTL transistor. They are useful for evaluation purposes, but they are not used to set test acceptance limits. The acceptance of an element is based upon measurements which give direct indication of sufficient current available rather than upon calculated values.

The Production Tester

As can be seen from the DCTL circuit considerations and from the foregoing graphical analysis, an adequate external production test of micrologic elements must consist of three basic types of measurements: input current, output current available, and output $V_{CE(sat)}$.

Definitions. The terms to be used in describing these tests are defined as follows:

1) $\overline{V}_{on}$ is the highest turn-on base voltage which will appear in any system. It is equal to V_{cc} applied through the lowest value of collector resistance.

2) $\underline{V}_{on}$ is the lowest turn-on base voltage allowed to appear in any system. It is derived in the same manner as the $\underline{V}_{on}$ that we discussed earlier.

3) $\overline{V}_{off}$ is the highest value of turn-off base voltage allowed to appear in any system. It is set high enough to permit an I_{CEX} comparable to that which will occur at high temperature.

4) $\underline{V}_{off}$ is the collector voltage applied during measurement of I_B. It is the lower limit of the voltage appearing at the collectors of three paralleled saturated transistors.

5) V_{sat} is the highest acceptable $V_{CE(sat)}$ for a transistor driven by $\underline{V}_{on}$ at room temperature. It is appreciably lower than $\overline{V}_{off}$.

With these definitions we can define the three types of tests performed by the production tester.

Input Current. With applied voltages of $\underline{V}_{on}$ at the base and $\underline{V}_{off}$ at the collector, the base current drawn by any transistor may not exceed a maximum value, defined as I_B. See Figure 12. This measurement is performed only on bases which are connected to micrologic element input terminals.

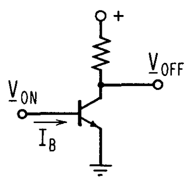


Figure 12.
Input Current.

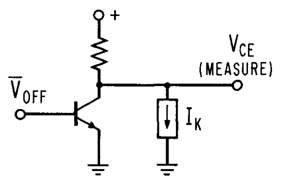


Figure 13.
Current Available

Current Available. This is a measurement of element output characteristics. With $\overline{V}_{off}$ applied to the base of an output transistor, and with current I_K drawn from the collector node, the collector node voltage must exceed $\underline{V}_{on}$. See Figure 13. If--as is often the case--the base is not accessible, then a combination of inputs is applied such that the highest possible turn-off voltage appears at the base.

$V_{CE(sat)}$. This is the other output measurement. With $\underline{V}_{on}$ applied to the base of an output transistor, the collector must pull down below V_{sat}. If the base is not directly accessible, then the worst-combination of inputs is applied such that the lowest possible internally-generated turn-on voltage appears at the base.

The loading conditions and the acceptance limits for these tests are set above those required to meet the specified fan-out of the elements. The margin of excess is one which has been empirically determined to insure operation with noise rejection over the specified temperature range.

Once the general test methods and criteria are established, the only remaining trick is to determine the worst-combination of conditions to apply at the other pins when performing a measurement at any given pin. Each element presents its own problems and requires a separate solution; the simplest element, "G," requires seven test steps; the most complex element, "S," requires 16.

It might be helpful to discuss some of the general considerations involved in the devising of the tests. The tests of input characteristics are rather straightforward, since only transistor bases are connected to the input pins; the only problem is to insure that the most severe collector-clamping conditions prevail. In the case of the "G" element, where the collector node is available and could conceivably have additional transistors connected in parallel, $\underline{V}_{off}$ is applied directly. See Figure 14. In the case of

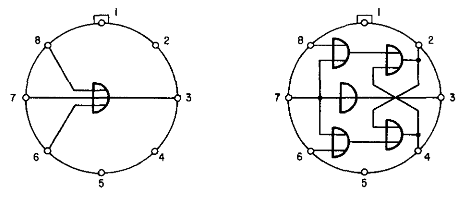


Figure 14. Logic Diagrams of the "G" and "S" Elements.

the "S" element, the input-gate collector nodes are not accessible, and the worst possible clamping for input current at pins 6 and 8, for example, occurs when pin 7 is driven by $\overline{V}_{on}$.

For current-available measurements at output terminals, a greater number of variables must be considered. All transistors whose collectors are connected to the given terminal must be turned off as weakly as possible in order to draw as much leakage current as will ever be experienced; all transistors whose bases are connected to the given terminal must have the worst possible collector-clamping in order that the bases will draw the maximum current. Then, with as much current as possible being internally drawn from the node, the external load current is drawn, and the node voltage checked to insure that it is higher than $\underline{V}_{on}$.

If a single transistor drives an output node, the test of $V_{CE(sat)}$ is made by turning it weakly on with $\underline{V}_{on}$ and measuring V_{CE}. If several transistors in parallel drive a node, then one is turned weakly on with $\underline{V}_{on}$ and the others are turned off with $\underline{V}_{off}$; the test is then repeated until each transistor has demonstrated its ability to hold the node voltage sufficiently low.

<u>Test Example</u>

As an example of the application of these general considerations, let's look at a specific micrologic production test, the test of the "F" element.

As summarized in Figure 15 and Table 1, the first two steps of the "F" test are measurements of the input base currents with the specified $\underline{V}_{on}$ applied at the base and with the collectors clamped. The currents are measured, the values recorded, and comparison made against the maximum acceptable value, I_B.

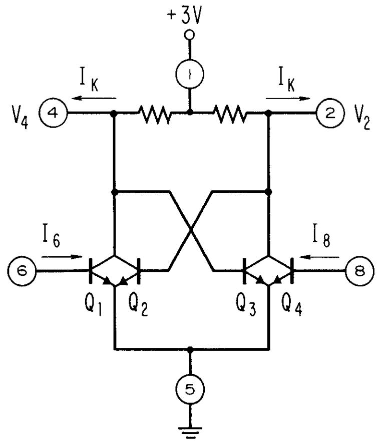


Figure 15. Schematic of the "F" Element.

In steps 3 and 4, minimum turn-on voltages are again applied to the "outside" transistors, Q_1 and Q_4, and their saturation voltage drops are measured and compared to V_{sat}.

TABLE I

Test	Apply	Measure & Compare	Remarks
1	$\underline{V}_{off}$ at pin 4 $\underline{V}_{on}$ at pin 6	$I_6 < I_B$	Check input current, Q_1
2	$\underline{V}_{off}$ at pin 2 $\underline{V}_{on}$ at pin 8	$I_8 < I_B$	Check input current, Q_4
3	$\underline{V}_{on}$ at pins 6 and 8	$V_4 < V_{sat}$	Check $V_{CE(sat)}$, Q_1
4	$\underline{V}_{on}$ at pins 6 and 8	$V_2 < V_{sat}$	Check $V_{CE(sat)}$, Q_4
5	$\underline{V}_{off}$ at pin 4 $\overline{V}_{off}$ at pin 8 Sink I_K at pin 2	$V_2 > \underline{V}_{on}$	Current available at pin 4, Q_2 clamped
6	Remove $\underline{V}_{off}$ at pin 4; other inputs remain	$V_4 < V_{sat}$	Check $V_{CE(sat)}$, Q_2
7	Same as test 6	$V_2 > \underline{V}_{on}$	Current available, Q_2 not clamped
8	$\underline{V}_{off}$ at pin 2 $\overline{V}_{off}$ at pin 6 Sink I_K at pin 4	$V_4 > \underline{V}_{on}$	Current available at pin 2, Q_3 clamped
9	Remove $\underline{V}_{off}$, all others same	$V_4 > \underline{V}_{on}$	Current available, Q_3 not clamped
10	Same as test 9	$V_2 < V_{sat}$	Check $V_{CE(sat)}$ of Q_3

In step 5, the collector of Q_2 is clamped, the maximum turn-off voltage is applied to the base of Q_4, load current is extracted from the node at pin 2, and the voltage at that node is measured and compared to $\underline{V}_{on}$. This step also serves to set the flip-flop to the proper state for the next two steps; Q_2 is now conducting and Q_3 is off.

Step 6 is a measurement of the $V_{CE(sat)}$ of Q_2.

Step 7 is another test for sufficient available current from pin 2. In step 5 we made this test with the base of Q_2 drawing as much current as possible from the node; in step 7 we test with

Q_3 drawing a maximum I_{CEX} from the node. Each of these situations is a possible worst-case; since they are mutually exclusive, two test steps are required.

Steps 8, 9, and 10 repeat the tests of steps 5, 6, and 7, but for the other side of the flip-flop.

Test Equipment. Two pieces of equipment have been built to perform the production testing described above; a small manual switching chassis has been built for low-volume testing of "F" elements, and an automatic system for the testing of all elements. The manual test chassis accepts voltages from four external power supplies and applies them to the device under test in accordance with the test steps shown in Table I. The current sink is furnished by a gounded-base 2N708. parameters to be measured are selected by the switch and applied to a VTVM for output indication.

This test chassis is shown in Figure 17. Shown at the right in this figure is the prototype of a device for semi-automatic insertion of micro-logic elements into circuit boards, adapter jigs, and breadboard sockets. Also shown is a micro-logic-to-octal adapter.

The Automatic Production Tester is a system for the automatic testing of all members of the micrologic family of elements. It performs high-precision tests with go, no-go comparison, digital read-out, and data recording.

Figure 16 is a block diagram of the test portion of the system; Figure 18 is a photograph of the system equipment.

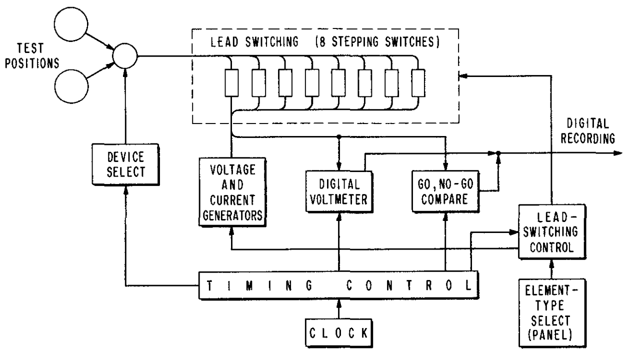


Figure 16. Block Diagram of the Test Section of the Automatic Production Test System.

Operation of the Automatic Production Tester. The leads of the elements under test are brought to eight stepping switches, which select the current or voltage to be applied or the measurement to be made at each lead. The stepping of these eight switches is under the control of the "lead switching control" section, which is programmed according to the type of element under test.

To perform a test, the operator first pushes one of six "select" buttons on the control panel to set up the program for the element to be tested. An element is then placed in one of the test positions and the "test" button pressed. The "timing control" section then takes over; this section consists essentially of a clock, a counter, and an order decoding matrix. The stepping switches then advance until "switching control" indicates that the correct conditions are set up for the first measurement. An analog go, no-go comparison is made, and the binary result signal is sent to the output unit. When the digital voltmeter has reached a null, its contents are sent to the output serial conversion unit, and while read-out is taking place, the tester procedes to the next measurement.

The output section of the system consists of a serial conversion and control unit, an IBM card punch, and a typewriter. All measured parameters are recorded; those that are outside the acceptance limits are indicated by an asterisk on the typewriter and a 12-punch on the cards.

Each measurement takes approximately one second, so that the complete test of an element requires from seven to 16 seconds, depending upon the element type. The tester has two test positions to permit the operator to remove a tested element and insert its replacement while tests are continuing at the alternate position.

Inter-relationship of Tests

We have been discussing the production test for some time, and it has been a while since we pointed out where it fits into the scheme of things. Lest the impression be conveyed that this test, as originally devised, is considered to be the final, ultimate, and definitive one, let us go back and look again at Figure 2. This flow diagram indicates that the production test, as now performed, is only our best first guess as to a test that is sufficiently severe without being unreasonably so. The efficacy of the tester on the production line is constantly being checked and second-guessed by the lab test and the production-tester in the life-test loop. The test parameters, criteria, and methods are subject to change if the information feedback indicates that changes are necessary to detect incipient failures.

Lab Test

The lab test consists of measurements of the r_b and h_{fe} of as many internal transistors as possible, of R_c wherever possible, of propagation delay time, and of the same characteristics as measured in the production test. These measurements are made for ambient temperatures from -55°C to +125°C.

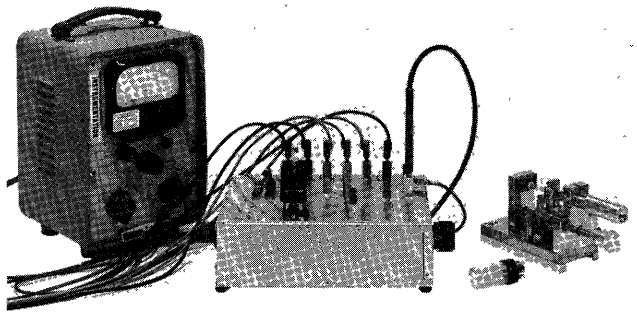


Figure 17. The Manual "F" Test Chassis.

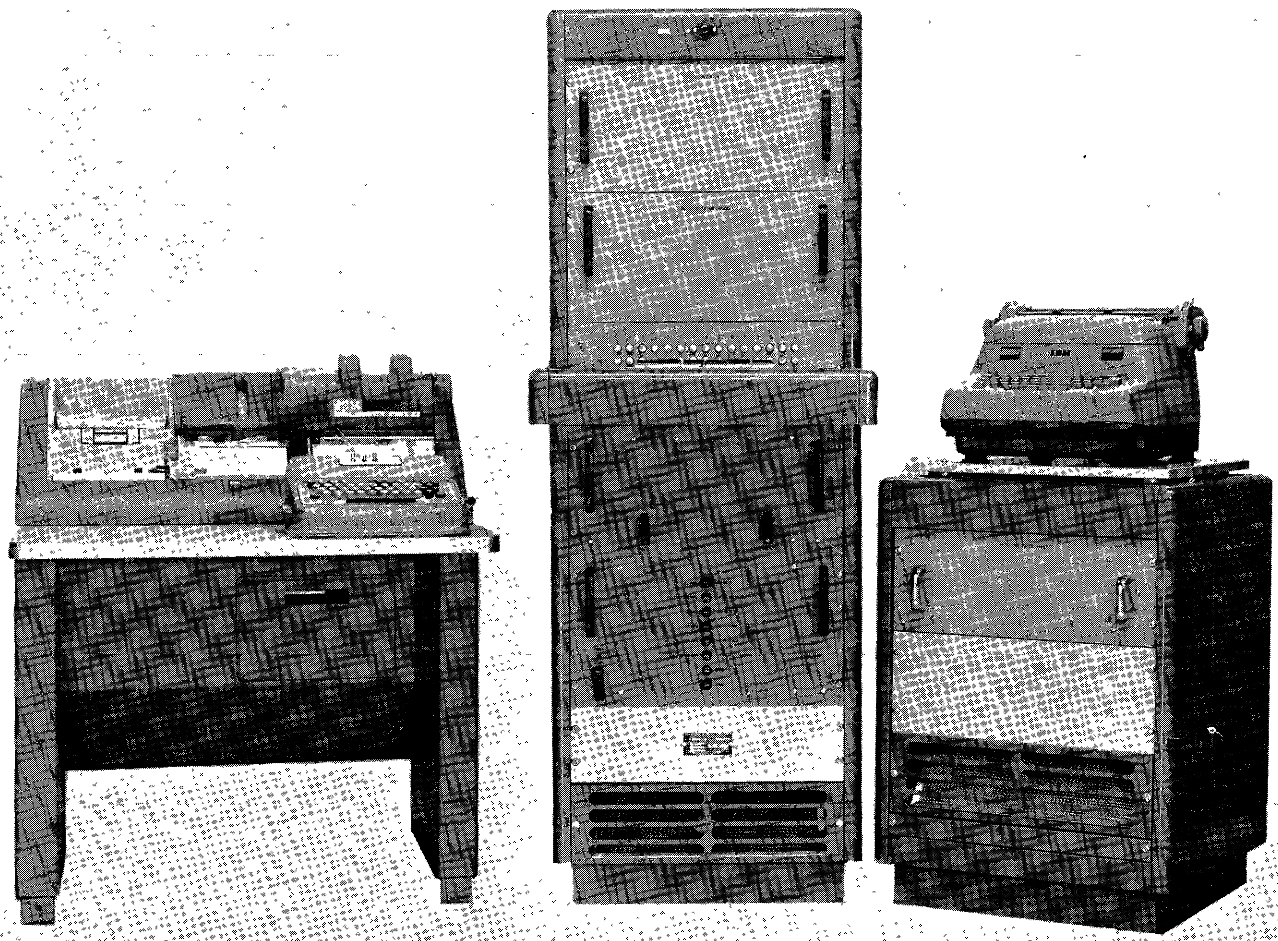


Figure 18. The Automatic Production Test System.

In Conclusion

The attempt here has been to demonstrate by example rather than by argument that valid evaluation tests can be performed upon a complex integrated functional circuit by operating only upon its external terminals. We assert that, when the element environment and use can be specified, it is more meaningful to evaluate the characteristics which are the resultants of the sum of the individual internal parameters than to evaluate the parameters themselves.

There is really nothing new in this. In the production testing of transistors in the past we have not been concerned directly with the base width, the minority carrier lifetime in the base, or the injection efficiency; we have been concerned only with the resultant sum of these parameters, as evidenced by the gain of the device. This is directly analagous to our measurement of current available at the output of a micrologic element, without regard to the gains or leakage currents of the transistors or to the values of the load resistors. In this sense, it could be said that the assertion we make here is no more than an extension of accepted practice.

Acknowledgments

Acknowledgment is gratefully made of the contributions of many people at Fairchild Semiconductor Corporation. In particular, the work of Richard Crippen in all phases of micrologic testing, and of James Nall and Lionel Kattner in the development of micrologic elements, has made this paper possible. Many suggestions of Donald Farina are incorporated in this paper.

Edward Russell, with Brian Bell and David Rollins, were responsible for the system, logic, and circuit designs and the fabrication of the Automatic Production Tester.

The support of the entire Fairchild organization has made the micrologic development program possible.

References

1. R. Norman, J. Last, and I. Haas "Solid State Micrologic Elements" presented at Solid State Circuits Conference, Feb., 1960.

2. R. Norman "Status Report on Micrologic Elements", presented at 51st Bumblebee Guidance Panel, June 22, 1960.

3. D. Farina, J. Nall, and R. Anderson "Application of Micrologic Elements", presented at National Electronics Conference, October 10, 1960.

4. E. Nussbaum, E.A. Irland, C.E. Young "Statistical Analysis of Logic Circuit Performance in Digital Systems" Proceedings of the IRE, January 1961.

INTERCONNECTION TECHNIQUES FOR SEMICONDUCTOR NETWORKS

J. S. Kilby
Texas Instruments Incorporated
Semiconductor-Components Division
Dallas, Texas

The semiconductor miniaturization approaches which have been described recently have promised complete electronic equipments of extremely small size, light in weight, and of high reliability. Although complete equipments have not yet been built from these devices, this paper will describe some of the factors which must be considered in equipment design and show one technique which might be used for high density equipment.

A typical unprotected semiconductor network is shown in Figure 1. This device is a flip-flop with sufficient gating to permit its use as a counter, shift register or set-reset flip-flop. In this design, two transistors are formed on square mesas near the center of the silicon bar. The material between the transistors forms the collector load resistors. The upper pair of arms extending from the center area are the cross coupling resistors, while mesa areas on these resistors provide the speed-up capacitors. The lower pair of arms is used as resistors on the gating networks. The four diodes required for gating are located along the lower edge of the bar. Two capacitors are formed on a separate bar, using the silicon oxide technique. Thermo compression bonded leads are used to make connections between areas on the upper surface of the bar and for some of the external connections.

In order to be useful, this device must be packaged to provide complete mechanical and environmental protection. The package must also include means for bringing electrical connections in and out of the device and some provision for removing heat from the device. The methods chosen to achieve these results will directly affect the interconnection of semiconductor networks to form complete equipments.

The interconnection technique to be used in and end equipment is ultimately determined by the equipment designer. Only he can determine the relative weights to be ascribed to the important factors of size and weight, cost, maintainability and reliability. Different weights of these factors have resulted in very different assembly techniques for radios, airborne computers, and hearing aids, for example. It is not likely that an universal technique will be developed to satisfy these widely different end objectives. This paper will describe a design where size and weight have been minimized at the expense of increased cost.

At some time in the future, it may be possible to fabricate entire equipments, or very large sections of equipments as a single unitary structure. This approach may be considered if self organizing systems which can tolerate large numbers of defective components can be devised, or if processing yields can be raised to a point very near perfection. At present, however, it is essential to build small groups of components which can be assembled to form the complete equipment.

Althought no exact figures exist, it is believed that the optimum complexity for the individual package is a single functional circuit such as a flip-flop, logic element or a gate. Selection of a functional block of this type permits performance testing of the finished unit, which is always desirable and sometimes essential. since not all of the individual components can be isolated for testing. The flip-flop shown in Figure 1 is near the upper limit for present circuit complexity. This package contains the equivalent of sixteen components.

The use of a package of uniform size makes it possible to connect the packages together with less wasted space between packages, although some space inside the packages is unused. For this reason, all of the digital networks which have been made to date have been packaged in the case shown in Figure 2. The size of this package has been chosen rather arbitrarily. Its rectangular shape permits ten leads to be brought out on the two long sides with a spacing of 0.047 inches. A flat shape was chosen to permit optimum heat transfer from the silicon wafer to the outside of the case.

This package is assembled by the process shown in Figure 3. This process provides a complete glass-to-metal hermetic seal, which is believed to be essential for full protection of the device under severe military environments. Because of the very small mass of the package, it is not susceptible to mechanical shock.

The thinness of the package makes it possible to connect packages together either by stacking or by the use of flat layouts on an etched circuit board. Since the thickness of the package is about equal to that of the common circuit boards, the volumetric efficiency of this technique is quite low. It does offer good access to the packages for testing and maintenance and should be quite useful in designs where minimum size is not a requirement.

For either the stacked configuration or the flat version, some form of multiplane wiring is probably essential. It is not possible to specify the lead sequence from the packages since the leads must be connected inside the packages to the device as directly as possible. The external wiring must therefore have some provision for crossovers.

One multiplane wiring scheme which has been used with success is shown in Figure 4. Here the packages are stacked, and thin sheets of teflon with metal cladding are used to form the conductors. It is frequently desirable to separate the supply voltage wiring, which may be connected to all packages in a stack, from the signal paths which go from package to package. One sheet may be used for each supply voltage. These sheets are formed with a grid patters of conductors and holes. The first sheet is placed over the leads of the stack and the leads to be connected to the sheet are bent over and soldered to the sheet. Electrical and mechanical clearances are provided so that the other leads will pass straight through the sheet and will be insulated from it. A second sheet may then be added and connected. Some of the stacks which have been built have used four supply voltage sheets. The signal paths which are required are then formed on similar etched sheets which complete the remaining connections.

An alternate type of construction is shown in Figure 5. The teflon sheets are quite similar to those used in the original version, but small flaps have been cut which can be bent to lie parallel to the leads with which they are to be connected. This version is particularly adaptable to welding. It has the added advantage that no bending of the leads is required and that all leads are available for use as test points after the stack has been connected.

Although defective packages have been replaced in stacks of this type, it would certainly not be attempted for field repair. The stack itself should be considered as the basic replaceable item. Although there is no single figure for optimum throwaway cost for present day military equipments, several studies have shown that the optimum is probably in the range of $200 to $500. It is believed that the cost of a ten to twelve package stack of networks will be within this range for production quantities of devices.

Since the stack is to form the replaceable element, it should be sturdy enough to withstand handling. It should also include a connector to permit easy replacement and isolation of the individual stacks for testing. One such arrangement which has been used is shown in Figure 6. An aluminum frame is used to hold the packages. The teflon sheets are used to provide connections between the packages. The ends of these sheets are then formed around the ends of the frames to provide the male portion of a connector. Flat side plates of aluminum are used on the frame to permit heat transfer from the stack. If required, aluminum foil strips may be placed between the packages and brought over to these plates to further reduce the temperature drop between the frames and the device junctions.

Stacks of this type which will accommodate twelve packages are 0.312 x 0.600 x 0.200 inches. Connections between stacks are provided by strips of connectors, which utilize a similar multiplane wiring scheme. A row of ten stacks is shown in Figure 7. The individual frame side plates are exposed so that the row can be sandwiched between thermal conductors. The edges of the multiplane wiring are again wrapped around an exposed edge of the strip to provide connections between rows.

These rows may then be plugged in to form large sections of an equipment or complete equipments, as illustrated in Figure 8. Multiplane wiring is used between the connector clips.

An assembly of 600 network packages is shown in the photograph of Figure 9. Although the finished equipment is to contain only 600 networks, a 20 per cent overage has been provided, or 720 possible package locations. These have been provided in six rows of ten stacks. Three rows are visible in the photograph, with the other three on the bottom of the package. Thermal mock-ups of the assembly have been completed and the preliminary data will be presented at the meeting.

The size of the finished unit is almost exactly that of a package of regular cigarettes. It would contain about 8500 individual components in the 600 packages. Total volume required is slightly under 6 cubic inches, including that required for the case, internal heat transfer provisions, and connectors.

This design is not believed to represent the smallest, or the lightest, or the cheapest version possible for this equipment. Many different arrangements of these parts are possible, and some of them may well be more desirable. Different objectives, in particular, may suggest radically different methods of construction. The real significance of this design is that of an existance theorem -- that it is possible to construct useful equipments from semiconductor networks which are orders of magnitude smaller than existing equipments.

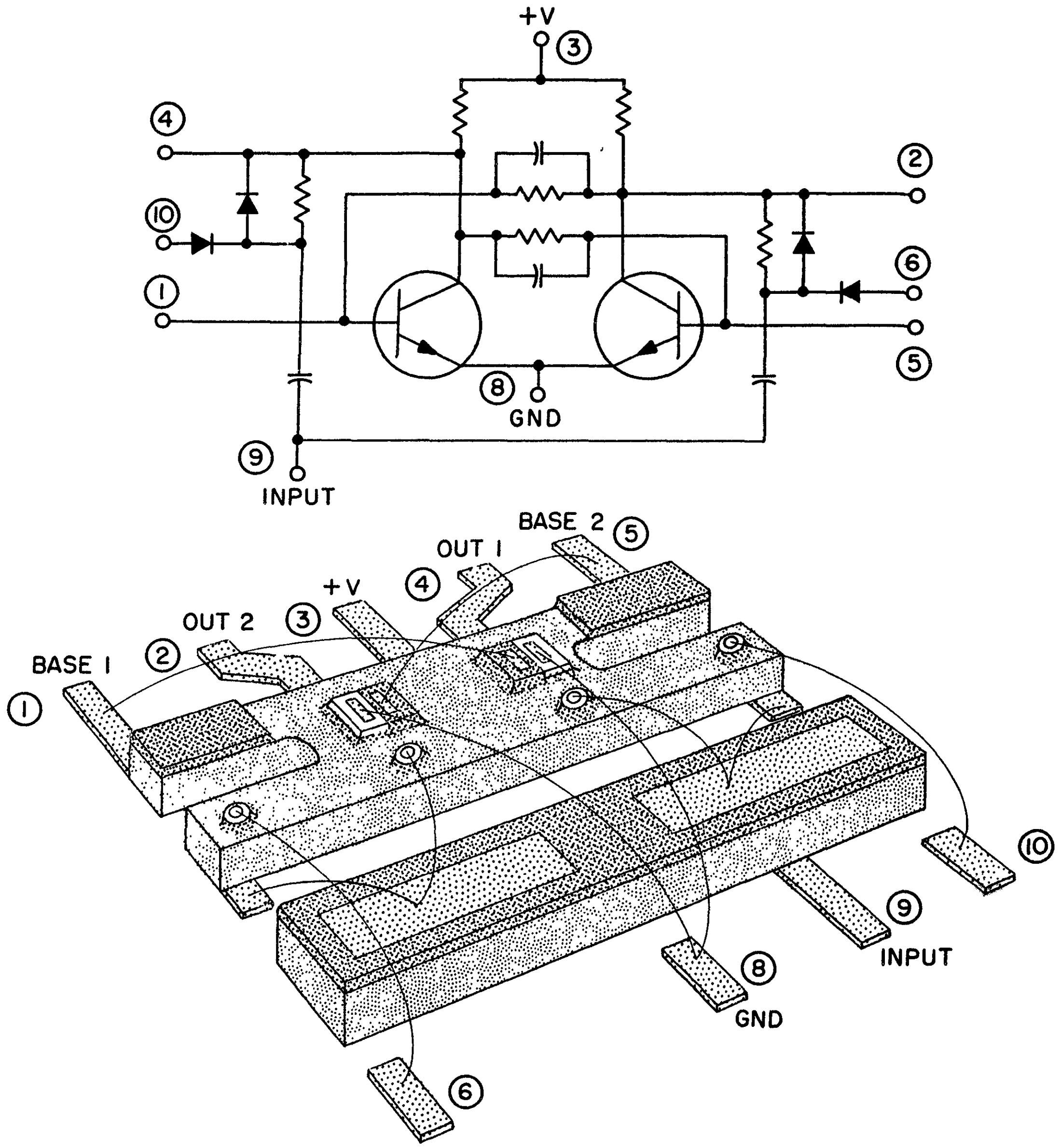

+V
3
4
10
1
2
6
5
8
GND
9
INPUT
BASE 2
OUT 1
+V
OUT 2
BASE 1
10
9
INPUT
8
GND
6
Layout Of Bistable Multivibrator (Type 502)
SOLID CIRCUIT semiconductor network

Figure 1 - Layout of Bistable Multivibrator (Type 502)
SOLID CIRCUIT Semiconductor Network

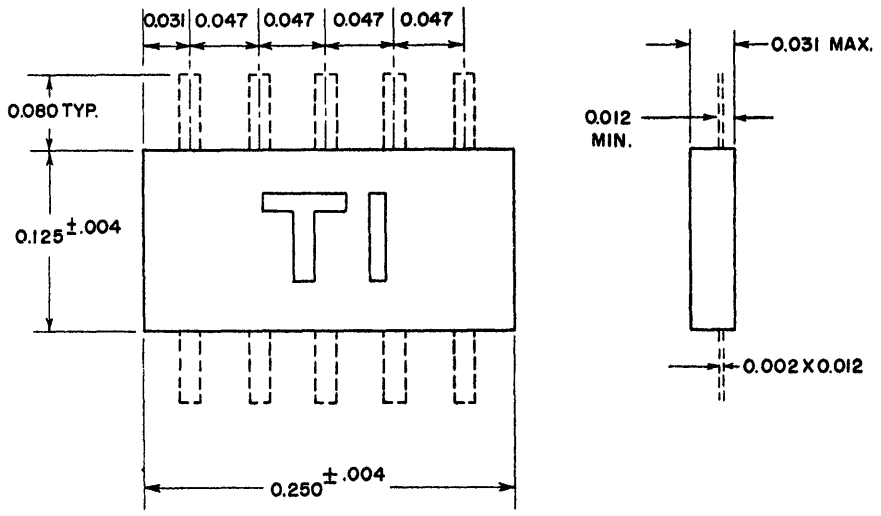
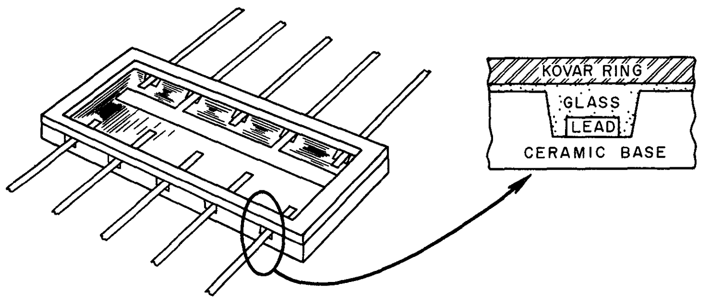


Figure 2 - Outline Dimensions, Hermetically Sealed SOLID CIRCUIT Semiconductor Network



SEMICONDUCTOR NETWORK HEADER

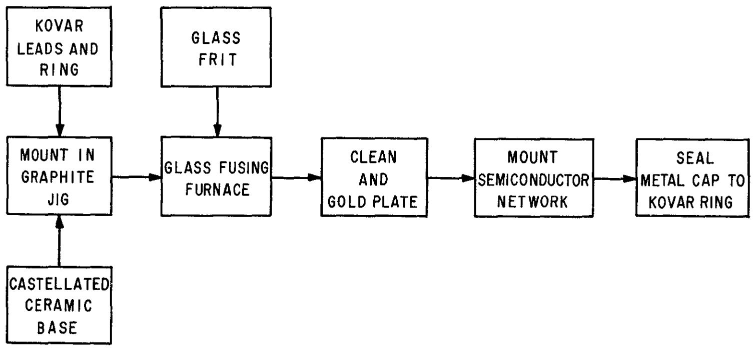
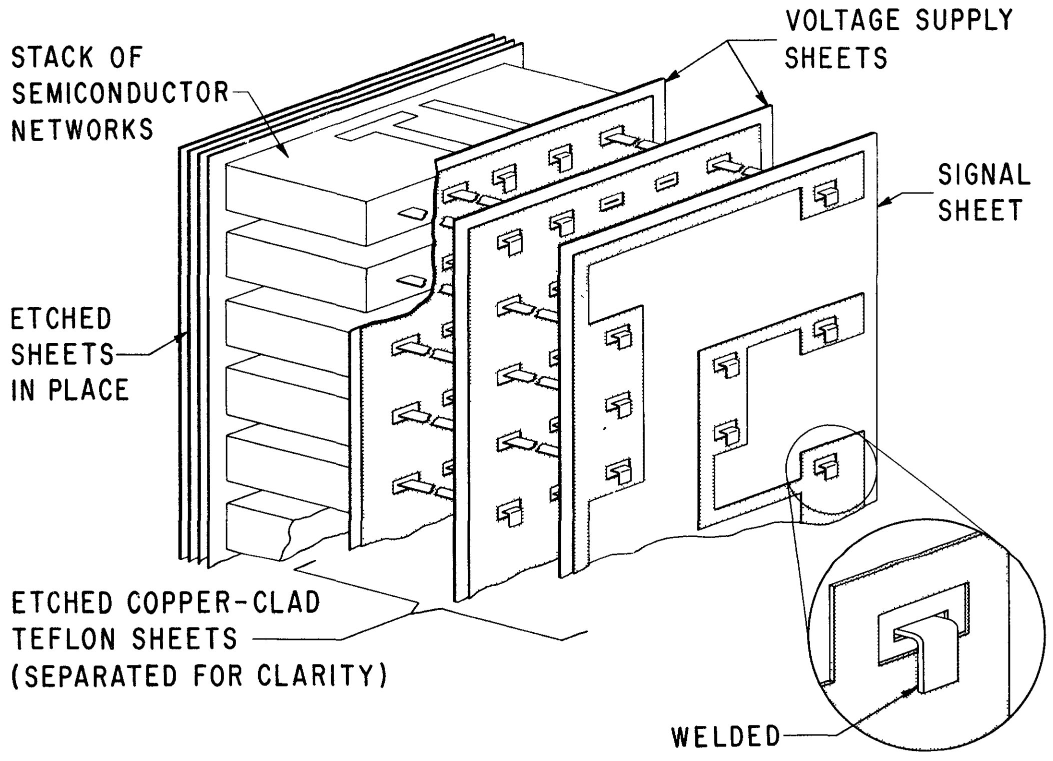


Figure 3 - Semiconductor Network Hermetically Sealed Package Assembly

PACKAGE INTERCONNECTION


STACK OF SEMICONDUCTOR NETWORKS
VOLTAGE SUPPLY SHEETS
SIGNAL SHEET
ETCHED SHEETS IN PLACE
ETCHED COPPER-CLAD TEFLON SHEETS (SEPARATED FOR CLARITY)
WELDED

SOLID CIRCUIT **semiconductor networks**

Figure 4 - Package Interconnection

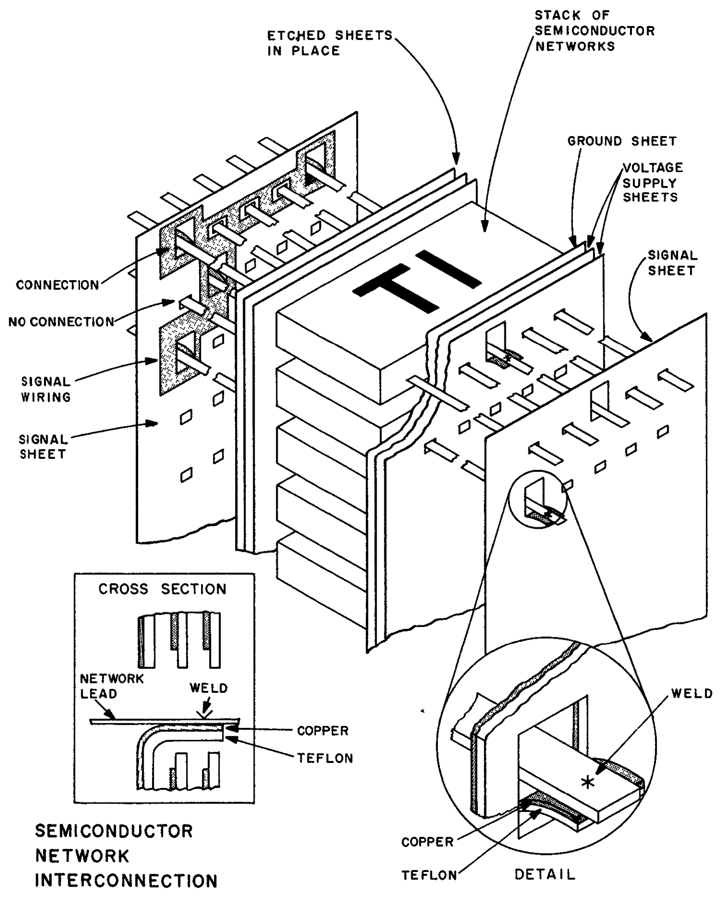


Figure 5 - Semiconductor Network Interconnection

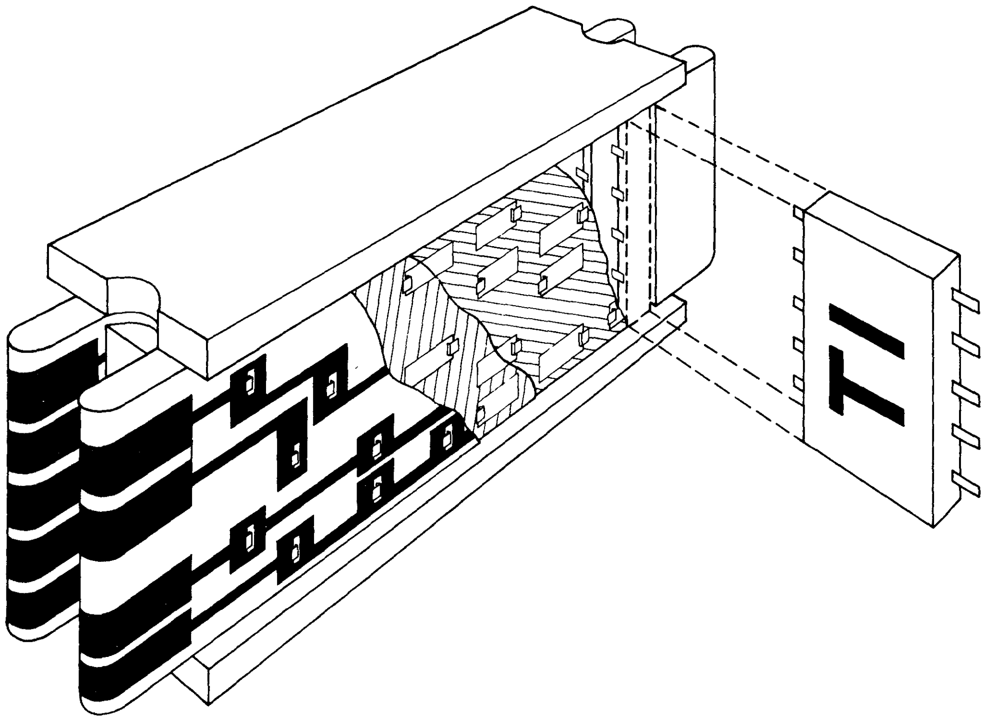


Figure 6 - Semiconductor Network Stack

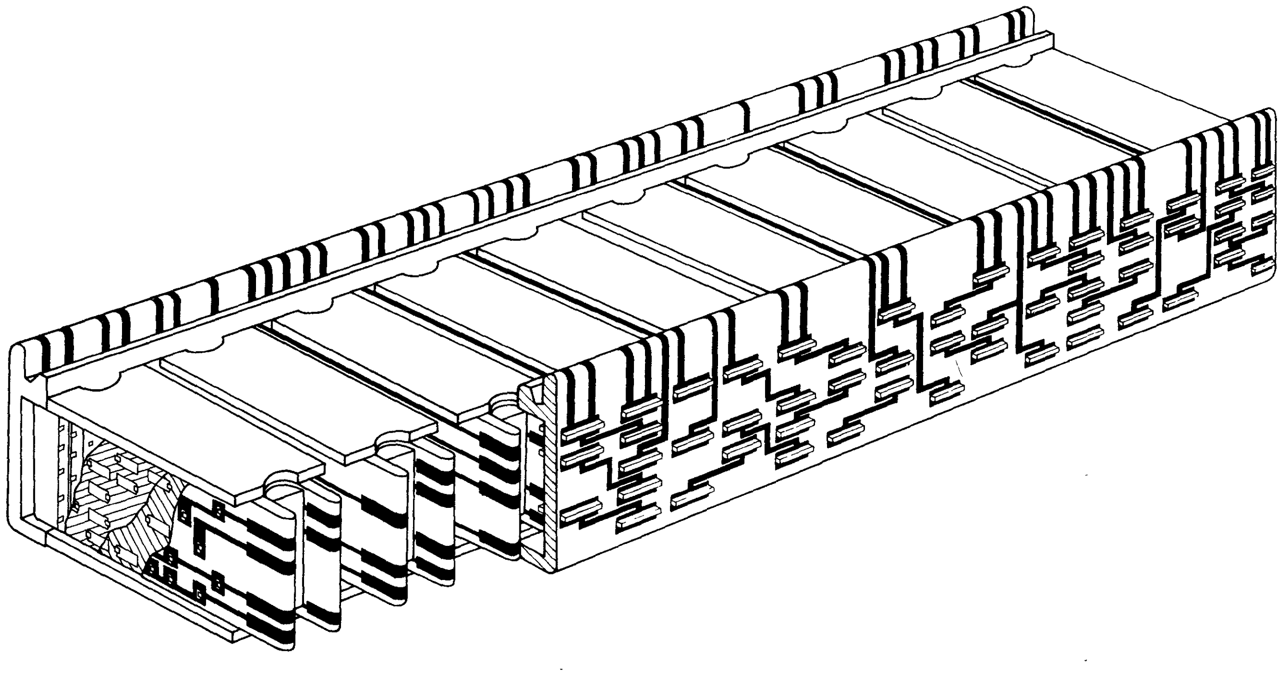


Figure 7 - Semiconductor Network Row

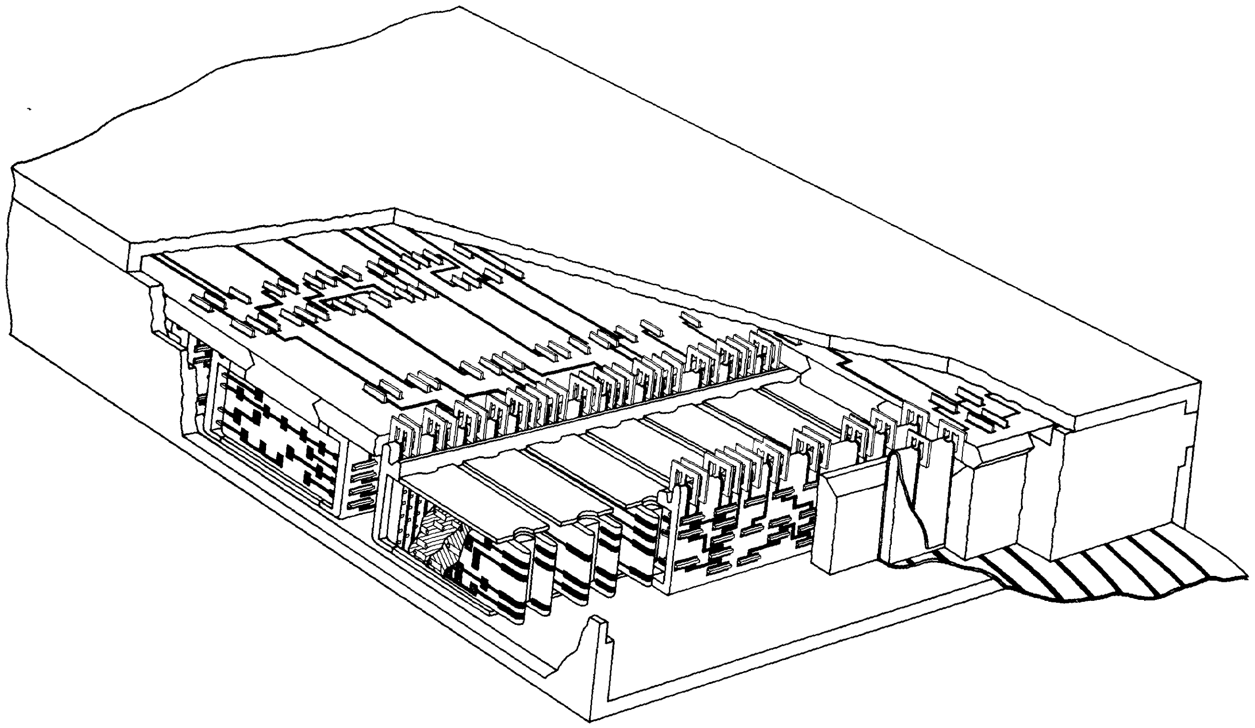


Figure 8 - Semiconductor Network Equipment Assembly

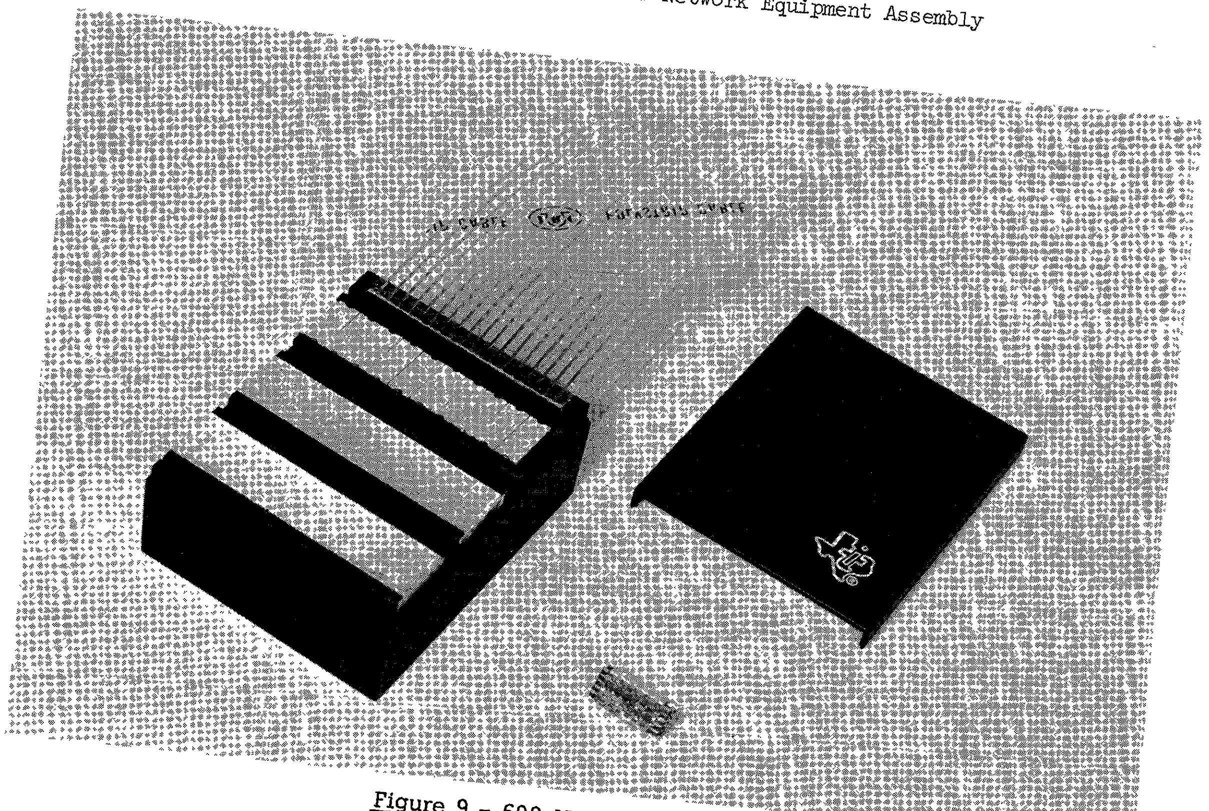


Figure 9 - 600 Network Assembly

MICRO-SYSTEM COMPUTER TECHNIQUES

by E. Luedicke and A. H. Medwin

Micro-Electronics Department
Semiconductor and Materials Division
Radio Corporation of America
Somerville, New Jersey

ABSTRACT

This paper briefly describes some of the problems encountered in building very high-speed (nanosecond) computer systems. A number of the techniques developed for a tunnel diode computer are described in detail, although it is emphasized that the difficulties are due to the operating frequency, rather than to any characteristic of the switching device.

Among the items discussed are ceramic circuit wafers with Fired-on metallized circuitry and ground plane; vacuum deposited rod resistors developed for this program; a grooved channel wiring assembly which holds the wafers; a new approach to a flexible transmission line; and a tunnel diode memory plane.

INTRODUCTION

As computers are made faster, the electrical requirements of the packaging system become more severe. Development of the Project LIGHTNING, 1000-megacycle, tunnel-diode computer has shown three main problem areas:

1. Wiring delays become significant.
2. Signal waveform distortion is greatly increased.
3. Signal crosstalk is greatly increased.

From the manufacturing point of view, the packaging scheme must be practical, which means relative ease of fabrication or, in other words, high yield. The wiring should also be flexible, so that leads may be opened, gates tied up, and signals simulated during the debugging phase.

High manufacturing yield and electrical flexibility indicate that the basic circuit unit should be small, since both yield and flexibility decrease rapidly if the basic unit consists of a large number of components. This is especially true if tight component tolerances must be maintained.

The geometrical shape of the basic circuit unit is primarily dictated by its electrical requirements. These must include not only the placement and interconnection of circuit components, but must allow a measure of freedom and ease in interconnecting a multitude of these basic circuit units. High component densities of a circuit unit cannot be utilized if its construction fails to provide electrical and mechanical interconnection flexibility to other units, a consideration which some microminiaturization schemes lack, and which cannot be sufficiently emphasized.

LOGIC CIRCUITS

Our present logic unit consists of a wafer with the tentative dimensions of 0.775 inch by 0.400 inch, as shown in Fig. 1. The material of the wafer is alumina, with a thickness of 0.020 inch. Terminal pads provided on three sides are used to connect the wafer circuitry with other similar units.

Utilizing three sides and a relatively high number of pads facilitates the layout of the circuitry on the wafer, since crossover of lines must be avoided. The back of each wafer is copper plated and is grounded. Metallized lines on the wafer are 0.025 inch wide and, with the dielectric of the wafer and its copper-plated and grounded back, form a transmission line (strip transmission line). Vacuum deposition of resistors directly on the wafer could be done since these techniques are well known. However, our circuitry requires one percent tolerances and this coupled with the fact that we may need six to nine resistors of three or four different values on a single wafer, places an unnecessary strain on experimental circuitry. Therefore, a ceramic rod resistor, 0.020 inch diameter by .100 inch long, was developed for the program. Each end of the resistor is metallized so that it can be soldered to the circuit pad; the resistive material is vacuum deposited on 0.060 inch of the body and covered with a deposited inorganic film and a silicone resin to avoid damage during handling and soldering.

An example of an actual experimental circuit is shown in Fig. 2. It is a dual-locked-pair circuit and consists of four tunnel diodes and eleven resistors. Reactive components are not required in this circuit. However, small inductances can be formed by loops. For inductance values which cannot be achieved by loops, aircoils could be used, but their utilization in high-speed circuitry is not extensive. Capacitors are soldered to the circuitry in the form of small ceramic pieces which have a high dielectric constant and are metallized on both sides. Since all components can be tested prior to assembly, the yield is high.

Consideration must be given to testing the assembled wafers at full operating speed. The test jig must provide electrically smooth connections between the test equipment and the terminal pads of the wafer. Fig. 3 shows such a jig. The wafer is placed in the opening near the top surface of the jig where spring contacts ground the copper-plated back of the wafer; the connecting wires are soldered to the terminal pads of the wafer and then guided in channels to coaxial connectors mounted at the rim of the jig. The test equipment is connected here using conventional coaxial cables. The wires in the channels and the channels themselves are designed to maintain a uniform impedance level and freedom from crosstalk.

To form an assembly with a multitude of wafers, provisions must be made to hold the wafer in place and provide the numerous interconnections. At high switching speeds, the requirement of suitable electrical interconnections between the wafers overrides all other considerations. These interconnections are no longer short in comparison to the wave length of the frequencies which they carry and, therefore have to be treated as a communication network; wiring delays, signal waveform distortions and crosstalk become significant design considerations. The final speed of the computer will be influenced by the speed with which this "communication network" can distribute the internal signals at the right time without major distortion and crosstalk to the individual circuits.

Printed circuit techniques have various electrical disadvantages which can usually be overcome in medium-speed computers but which cannot be tolerated in a high-speed (nanosecond) machine. On printed transmission lines, there is appreciable crosstalk between closely spaced parallel wires since the ground plane structure which is needed to electrically separate such circuits is prevented from doing so by the dielectric sheet. Crossovers of printed circuit lines are complicated and introduce electrical discontinuities and additional crosstalk. Non-uniformities in the wiring or associated fittings may cause spurious resonant modes in the relatively large dielectric sheet.

A shielded coaxial transmission line has none of these disadvantages; it provides a physical path with uniform inductance and capacity per unit length and has minimum crosstalk.

This transmission line concept is used in the channel wiring assembly. The basic idea is to place an insulated wire in a metal channel of a slab-like structure having two or more channel patterns. The same channels also are used to hold the wafer in place. The terminal pads of the wafer line-up with these channels, so that each wire remains in its channel until it reaches the proper wafer pad. The electrical connection of the wire to the wafer is done by stripping the wire insulation and soldering the wire to the terminal pad on the wafer.

The diameter of the conductor, dielectric constant and diameter of the insulation contribute to the characteristic impedance of the line. With each wire in its own channel, crosstalk is minimized. Further shielding can, however, be achieved by painting the top of the channel with conductive epoxy. Where the same pulse arrives at separate circuits at different times because of propagation delay, both wires can be precut to the longest length and the slack to the closest circuit taken up by running it back and forth in channels.

The channels are arranged in a 0.050-inch pattern which permits flexible wiring and still maintains transmission line characteristics. Where wires must cross each other, a hole is drilled in the channel and one of the wires goes through this hole into a channel on the reverse side and returns at a convenient place.

All the channels which run rectangular to the wafer plane have twice the depth of the channels parallel to this plane; this arrangement permits a wire to pass the wafer at the bottom or one of the two sides and go to wafers in the same row without leaving its own channel. A sketch of this is shown in Fig. 4.

A frame of a channel wiring assembly for three rows of wafers is shown in Fig. 5. The individual parts of this frame are cast in epoxy, chemically plated and then electro-copper plated to a suitable depth. This is a very economical process since only the masters are machined in brass and then cast in silicon rubber to form the mold for the epoxy pieces. Fig. 6 shows the brass masters, the silicon rubber molds, the cast epoxy pieces and an assembled, copper plated channel wiring unit.

To evaluate the properties of the transmission lines, channels of different widths and cross-cut patterns were machined in a 65-cm long epoxy piece which was then copper plated. This test arrangement, which is approximately ten times longer than the actual channel wiring assembly, was chosen to increase the accuracy of the measurements. Several types of wires were placed in these channels and measurements were made of the characteristic impedance, loss and crosstalk. These measurements were compared with the measurements in a solid piece of silver-plated brass of the same configuration. No significant difference was found between these two pieces. These test transmission lines are shown in Fig. 7. Crosstalk was measured by placing wires in two adjacent channels as seen in Fig. 8. In a single-cut channel, crosstalk was sufficiently reduced (-55db) such that covering the channels was not necessary; however, in the cross-cut channel, silver epoxy or silver paint was required to cover the channels in order to reduce the crosstalk to a value comparable to that obtained in single-cut uncovered channels. Fig. 9 shows an example of crosstalk measurements over the frequency range from 1.75 to 2.1 kilomegacycles.

The impedance of an ideal coaxial line, having a 23-mil O.D. "Teflon"* insulation and an 8-mil diameter conductor, was calculated to be 45 ohms using 2.0 as the dielectric constant of "Teflon". In the uncovered line, the inductance will be higher, and the effective k of the dielectric will be lower due to the uncovered top portion of the line. Using the measured k of 1.67, the calculated impedance is 56 ohms as compared to the measured impedance of 53 ohms. When the line is covered with conductive silver epoxy, the effective

*A registered trademark of the E.I. DuPont Co.

k is 7.5% lower than the value of 2, resulting in a calculated impedance of 46.6 ohms. This compares favorably to the measured value of 46.4 ohms.

In calculating the losses in the coaxial line for this condition, it was assumed that the top half of the outer conductor was covered with silver paint and the bottom half of the outside conductor was copper. Using this assumption, the losses in the epoxy line covered with silver epoxy should be 1.66 times more than the uncovered case, which were measured at 1 and 2 kilomegacycles to 0.51 and 0.77 db/ft, respectively. These calculations were made using a value of 30 x 10^{-4} ohm/cm as the resistivity of the silver paint and experimental results check the factor of 1.66 closely.

If this value of resistivity is accurate, the losses in the cross-cut lines should be 2.4 times higher than those in the coaxial line. Again the experimental values of loss in the cross-cut lines covered with silver paint are approximately 2.4 times the calculated values of a coaxial line.

Insertion loss was measured by plotting points on a Smith chart corresponding to a minimum position on the slotted line and a VSWR reading determined by moving a variable reactance in series with the line in test. Fig. 10 shows the test setup. This method finds the essential insertion loss which is unique for the line being tested. The insertion loss for the line being tested depends on the type of measuring line used. This particular method ignores the reflections at connectors along the line. Therefore, the essential insertion loss, which should be a smooth line as a function of frequency, is slightly less than the actual insertion loss if found by the substitution method. This method also assumes that the variable reactance is lossless. Calculations show that at 1 kmc, the loss in the variable reactance is 0.09 db. When a sufficient number of points are located, a circle can be drawn through the points. This circle can now be rotated so that it lies with its center on the resistance axis of the Smith chart. The equivalent VSWR can be calculated by finding the two values of VSWR at the points where the circle crosses the real axis. If the circle contains the origin of the Smith chart, the equivalent VSWR is found by:

$$\mathrm{VSWR}_{eq} = \left[(\mathrm{VSWR}_2)\ (\mathrm{VSWR}_1)\right]^{1/2}$$

If the circle lies outside the origin of the Smith chart, the equivalent VSWR is found by:

$$\mathrm{VSWR}_{eq} = \left[\frac{(\mathrm{VSWR}_2)}{(\mathrm{VSWR}_1)}\right]^{1/2}$$

After the VSWR is found, the loss can be found by:

$$\mathrm{db} = 10 \log \frac{\mathrm{VSWR} + 1}{\mathrm{VSWR} - 1}$$

Figure 11 shows a Smith chart with the losses calculated using the above described method.*

*This method was first conceived by D.R. Crosby and makes loss measurements possible without knowing the disturbing influence of connector cables and fittings.

Using a 23-mil wide channel and varying the ratio of conductor to insulator, impedances ranging from 140 ohms to 20 ohms were measured.

The upper limit of characteristic impedance was found by running a 2-mil conductor through a 23-mil outer diameter sleeve of "Teflon". The lower limit was obtained with a 22-mil conductor having approximately a 1-mil coating of "Teflon".

A line with a characteristic impedance of approximately 1 ohm was also built. This line was not a wire as such but a strip of ceramic with a dielectric constant in the vicinity of k ≈ 4000; it was 10-mils thick, 75-mils wide and 6 to 8-cm long. The high-k material was covered with conductive material on both sides (Fig. 12). This line is intended to be placed in a 75-mil wide channel (Fig. 13). Since the high-k material is very brittle and cannot be bent, its use will be confined to a row of wafers in the channel wiring assembly where the same termination point must be parallel connected to a low-impedance transmission source.

MEMORY CIRCUITS

The packaging requirements of a tunnel diode memory are in some respects more severe, and in other respects easier, than the logic wiring. The easy aspect is that the geometry of each memory plane is fixed and rigid so that the random wiring capability of the logic section is not required. The complication arises from the relative complexity of the basic storage circuit and from the limit placed on the propagation delay due to the high-speed operation, which in our case requires that a 32 by 32-bit memory plane does not exceed a volume of 3.5 inches x 3.5 inches x 0.1 inch.

As to be expected in the construction of a high-speed memory of this type, it is necessary that the bit and word lines must be transmission lines and that ground current paths must be carefully considered. The remaining part of the physical construction is influenced largely by the size of the components and electrical operation of the storage circuit.

We have been concerned with two different storage schemes: one is bit-organized storage and the other is word-organized storage, as shown in Figures 14 and 15, respectively.

The memory plane construction for the bit-organized storage, shown in Fig. 14, consists of metallized alumina sheets which are fired together. The bottom layer is metallized to form a ground plane on one side and 32 tapered lines are deposited on the top side.

The next layer has cut out slots to provide access to the bottom plane. Only its top is metallized. The third layer is also slotted; its top is metallized to form 32 tapered conductors.

After the three layers are registered together and fired, a 30-mil thick single-homogeneous plane is achieved. The slots give access to the 32 transmission lines on the bottom layer.

The next layer is a molded epoxy piece which is 100-mils thick. It is metallized on its top and has 1024 cut outs, one for each storage bit. The round portions of the cut out hold the two resistors which are required for this storage circuit. One resistor is physically longer than the other, but both have the same resistance value. The longer resistor is connected to the transmission line on the bottom layer and the short one to the transmission line which is on the top of the third layer. The tunnel diode is soldered with its anode to the tops of the two resistors and with its cathode to the metallized top of the epoxy piece. Another alumina sheet, metallized on its top, is placed over the anode leads of the tunnel diodes. It acts as a coupling capacitor for all storage circuits and connects to the sense amplifier. Since it is known which bit is interrogated, there is no problem of identification when an output pulse appears. An exploded view of this construction is shown in Fig. 16.

Another type of construction was chosen for the word-organized storage scheme (Fig. 15). While it has the same number of components for each storage bit as the first one, it differs by the requirement that it must be connected to a power supply bus, and that this bus must have the characteristic of a low-impedance transmission line.

A slab of copper-laminated "Teflon" was milled to produce 32 bit lines, each with an unloaded characteristic impedance of 65 ohms. The "Teflon" is 0.020-inch thick and the width of each line is 0.010 inch. The lines are on 0.100-inch centers as shown in Fig. 17. Notched copper bars, 0.010-inch thick, bridge each of the bit lines and permit a common ground at each storage bin. A 0.050-inch wide copper-laminated mylar strip is soldered to each side of the bar. With 1-mil mylar insulation, a characteristic impedance of 2.5 ohms is obtained. One of these lines is the word line, the other serves as a low-impedance supply bus (Fig. 18). The full view of the memory plane with its three lines is shown in Fig. 19.

A molded epoxy stick holds in position the 96 components required for 32 bits and is used as a subassembly as seen in Fig. 20. It is placed between the word lines. The cathodes of the tunnel rectifiers are soldered to the bit lines, the cathodes of the tunnel diodes to the word lines and each resistor is soldered to the supply bus with a jumper.

* * * * * * * *

Acknowledgement is gratefully made for the guidance of D.R. Crosby, the assistance of F. Borgini, the many contributions of J. Schiller and L. Trager, and the skills and knowledge of a host of RCA associates too numerous to list.

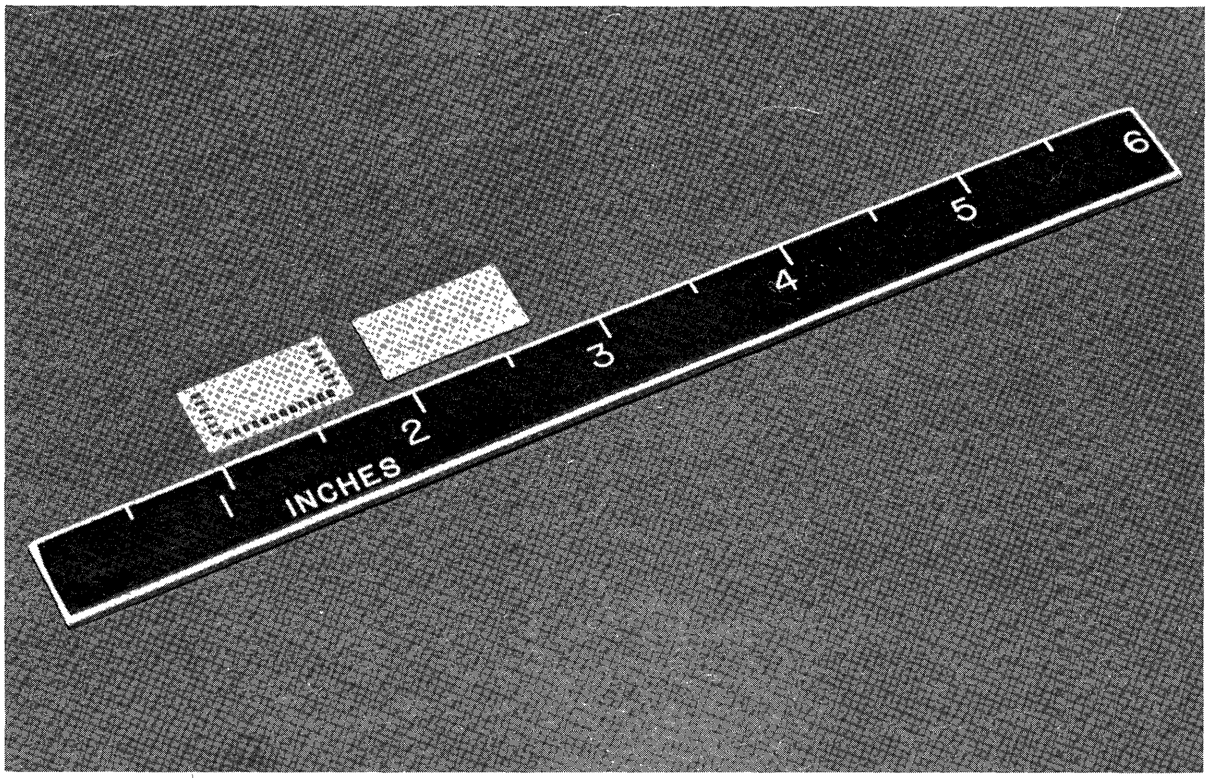


Figure 1. Alumina Wafer

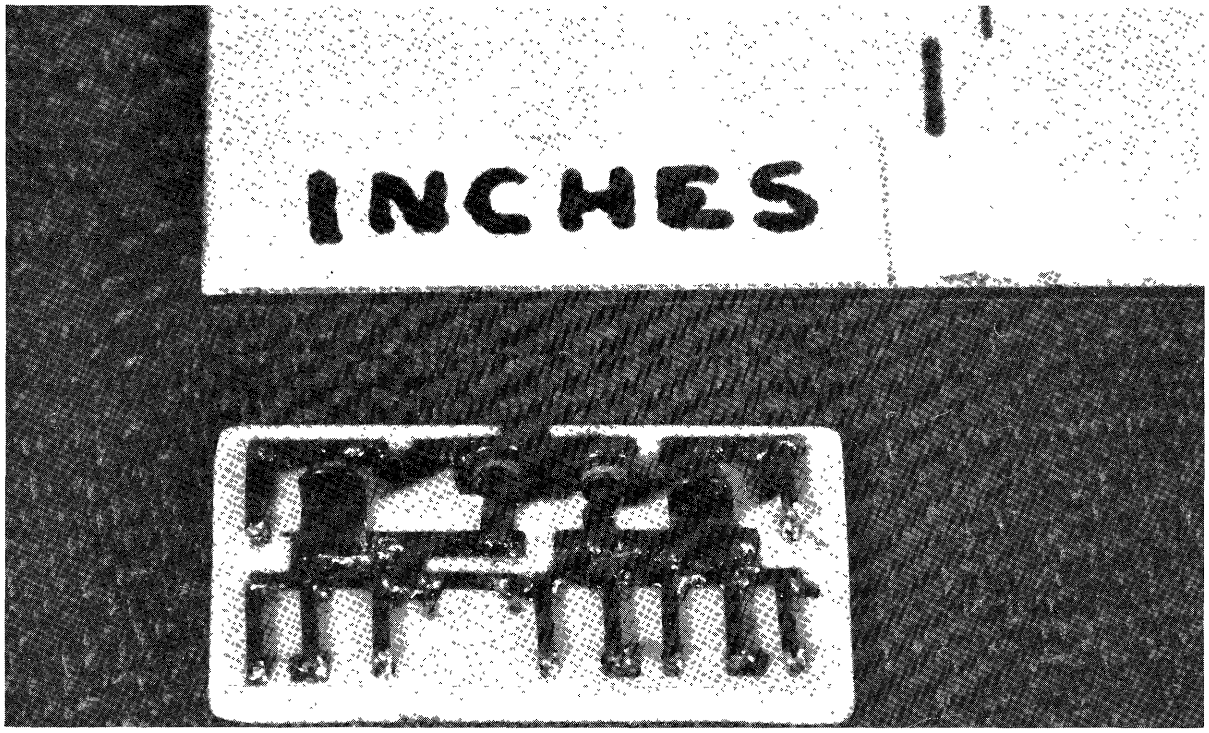


Figure 2. Dual Locked-Pair Circuit on Wafer

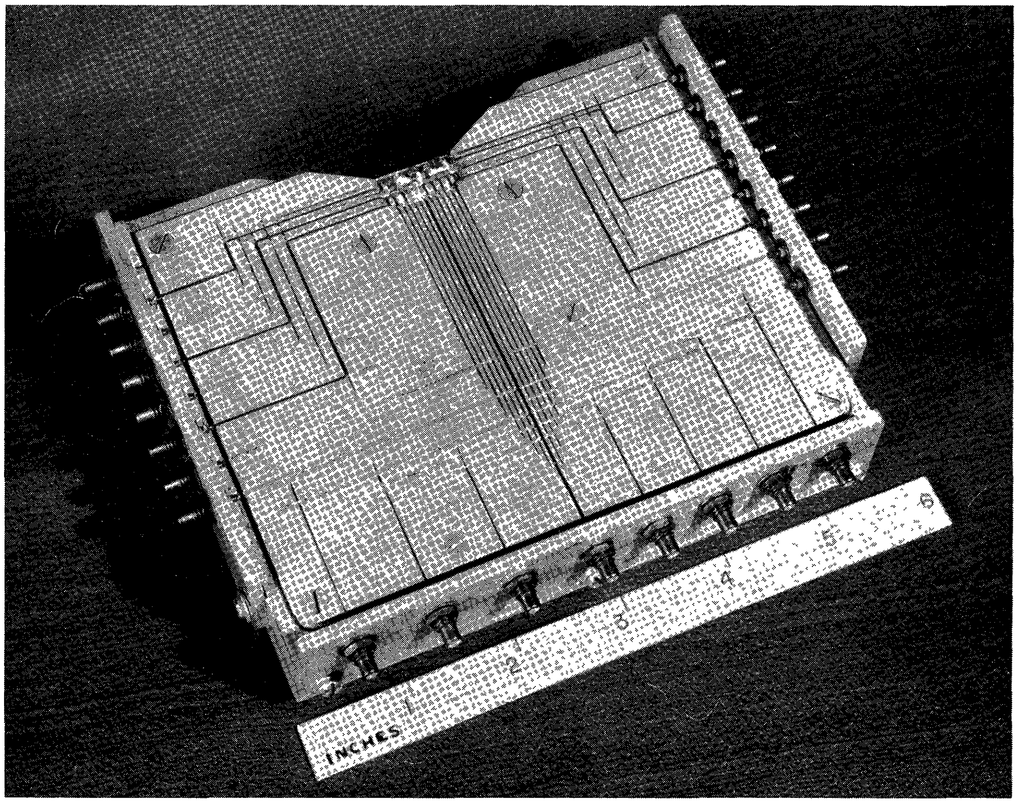


Figure 3. Wafer Test Jig

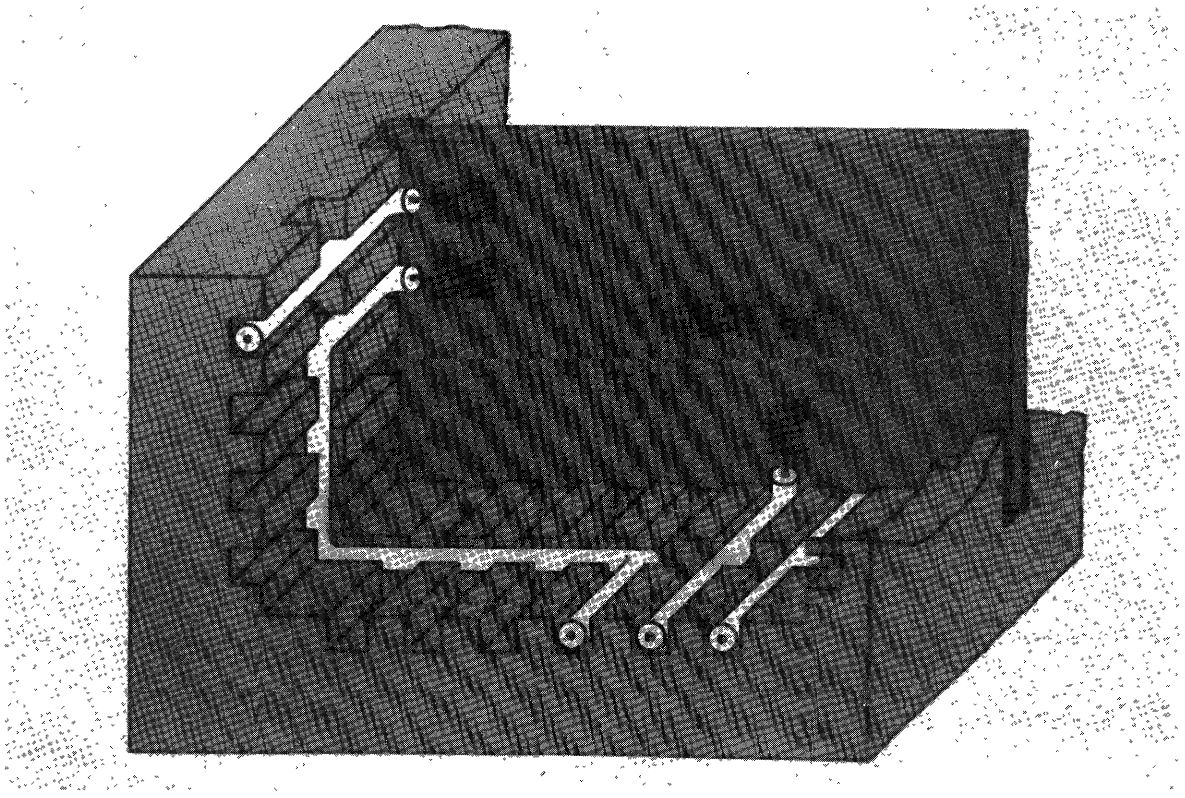


Figure 4. Wafer Interconnection With Transmission Lines

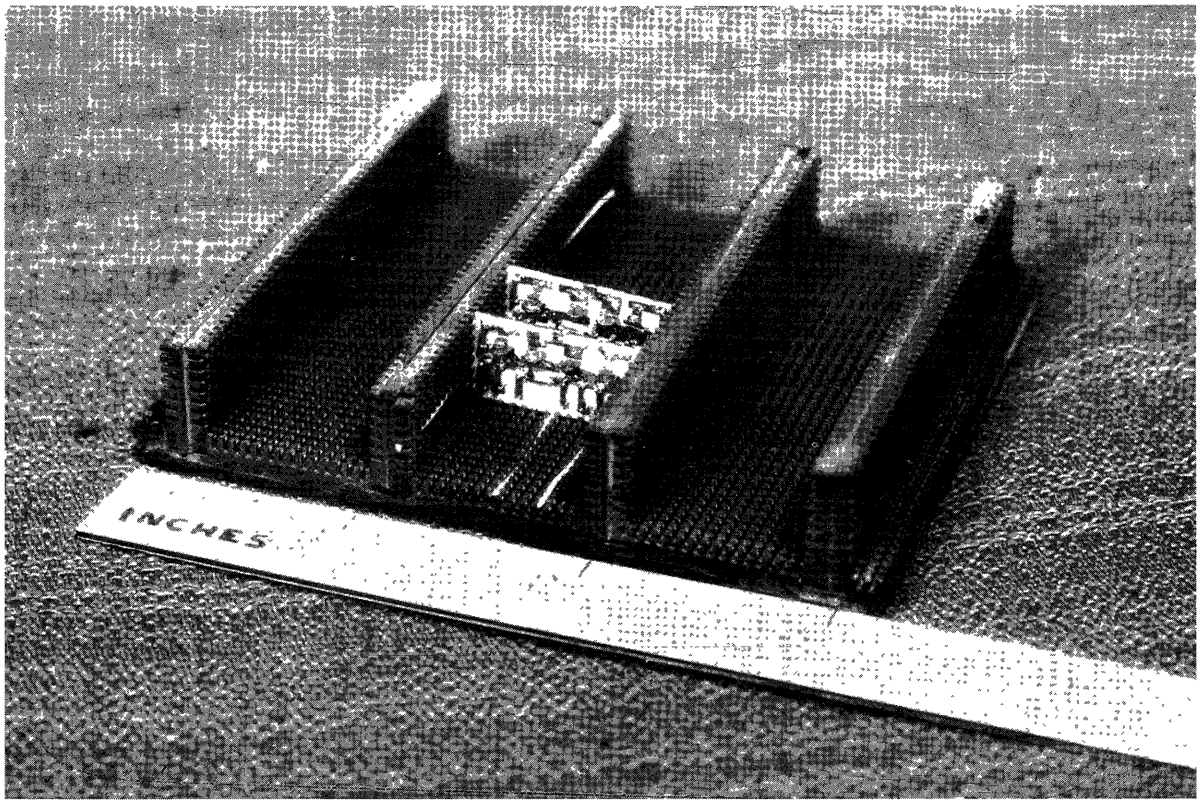


Figure 5. Channel Wiring Assembly With Two Wafers

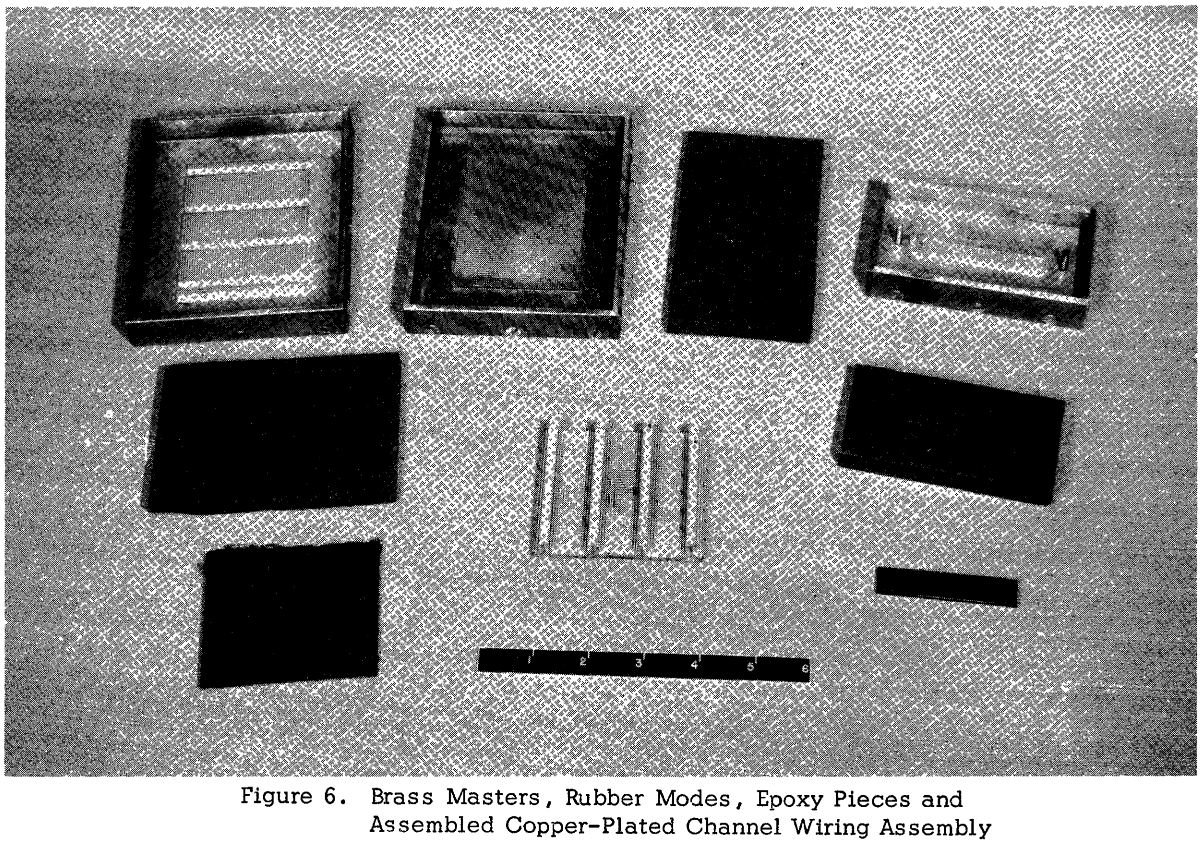


Figure 6. Brass Masters, Rubber Modes, Epoxy Pieces and Assembled Copper-Plated Channel Wiring Assembly

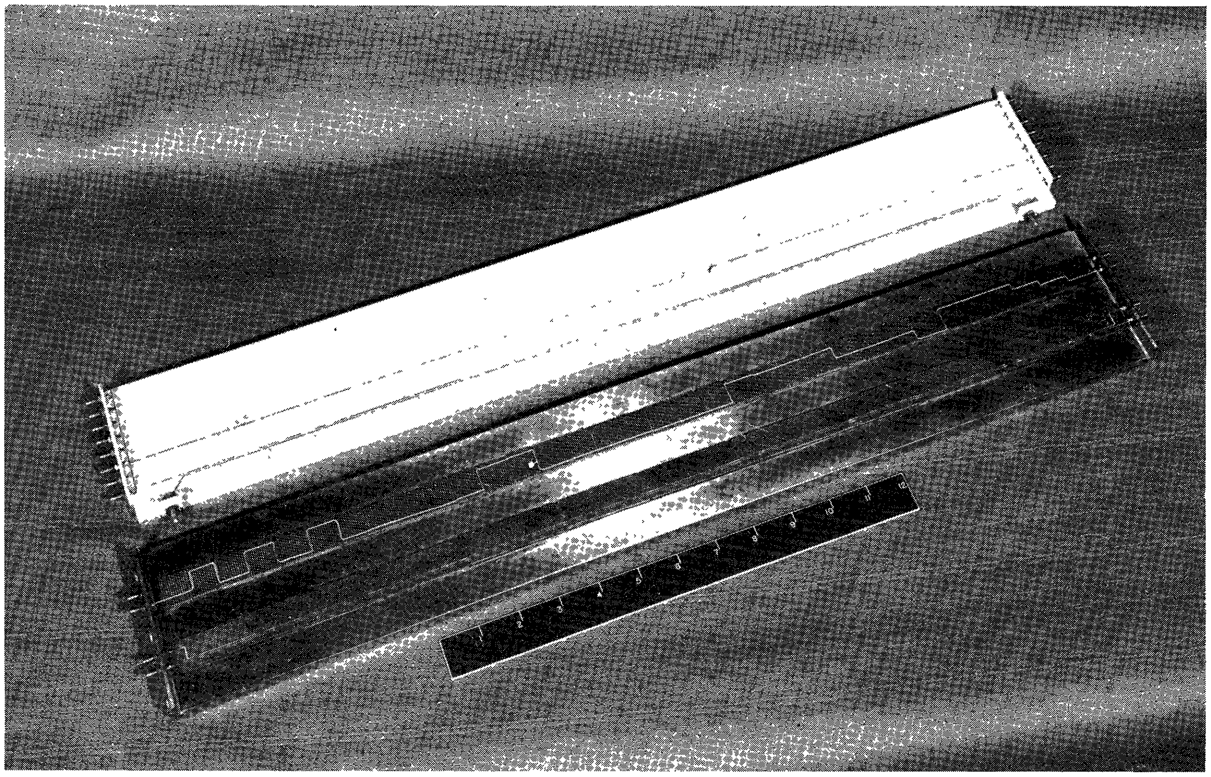


Figure 7. Transmission Test Lines

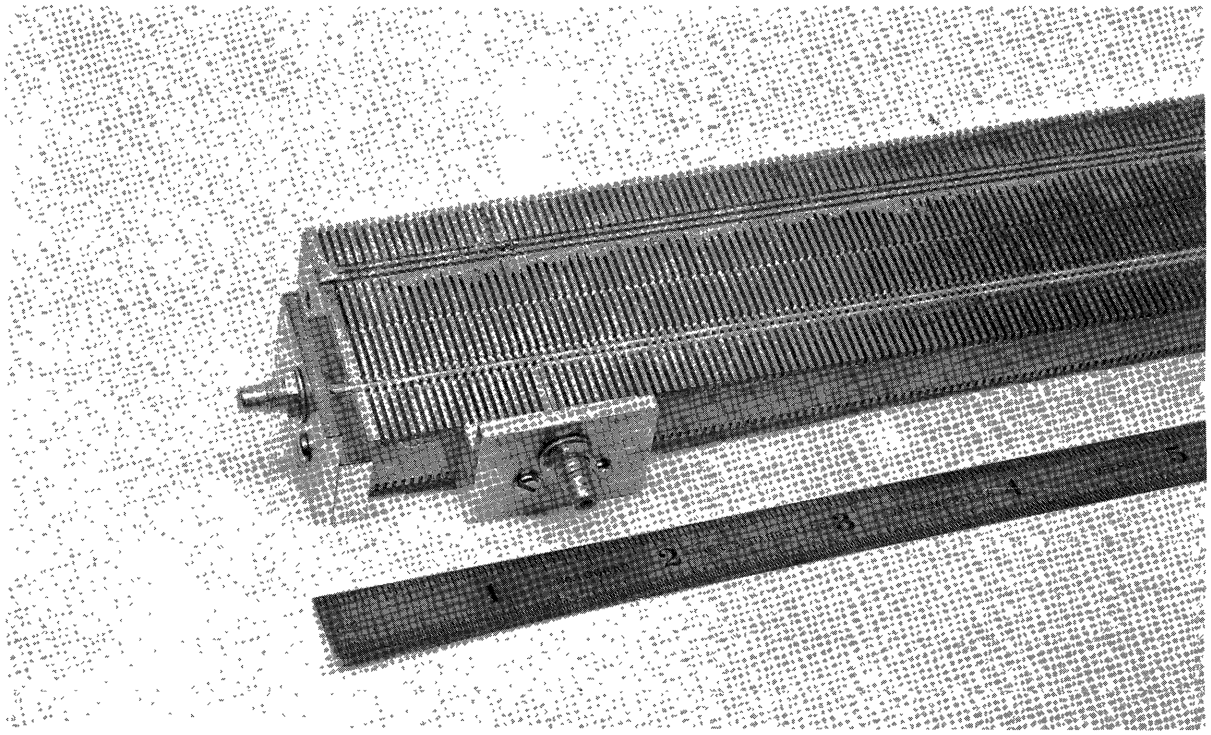


Figure 8. Part of Test Transmission Line for Double-Cut Channel Crosstalk Measurements

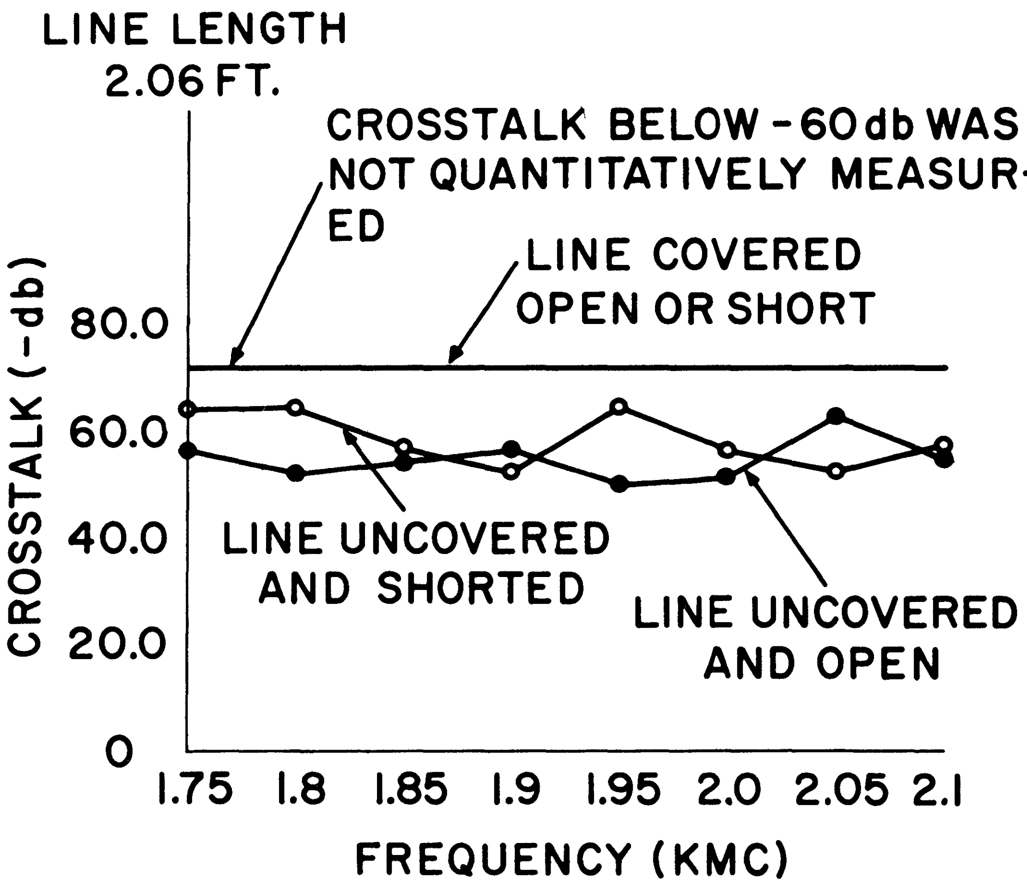


Figure 9 CROSSTALK VS. FREQUENCY OF COVERED AND UNCOVERED TRANSMISSION LINES

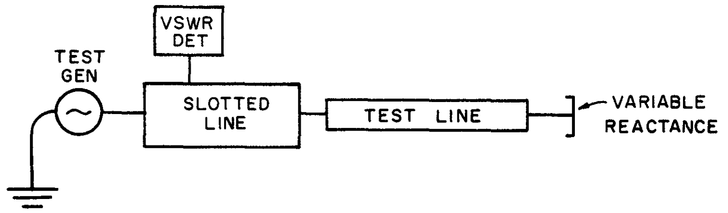
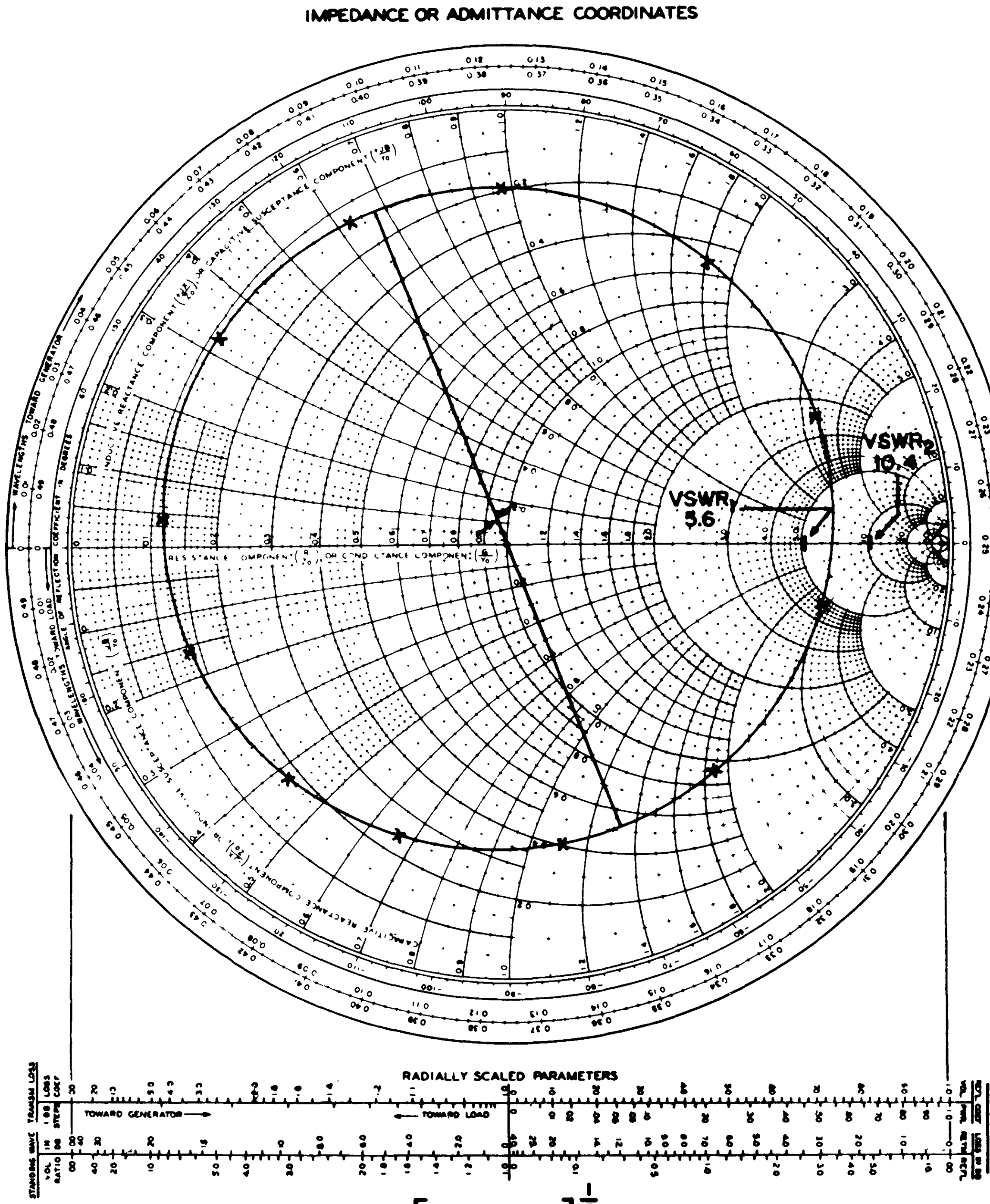


Figure 10 TEST SET UP FOR INSERTION LOSS MEASUREMENTS

FREQ. 1 KILOMEGACYCLE
LINE LENGTH 2.06 FT.
CHANNEL, BRASS .025" X .030"
TEFLON INSULATED WIRE .025" OD



$$VSWR = [(5.6)(10.4)]^{\frac{1}{2}} = 7.63$$

$$db = 10 \log_{10} \frac{8.63}{6.63} = 10(0.114) = 1.14\ db$$

Figure 11. **ESSENTIAL INSERTION LOSS**

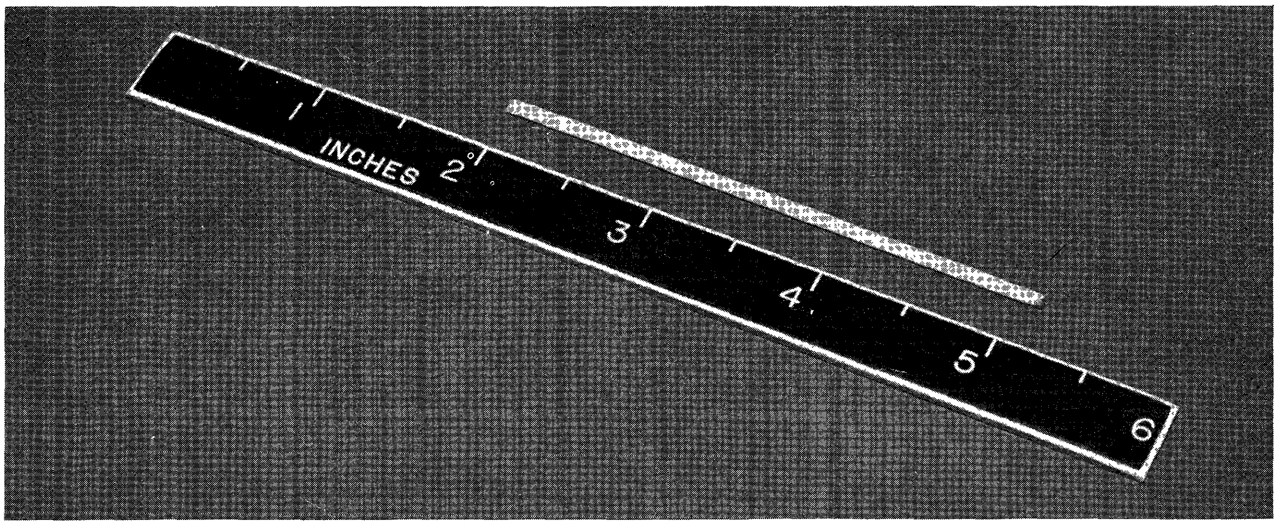


Figure 12 CERAMIC TRANSMISSION LINE

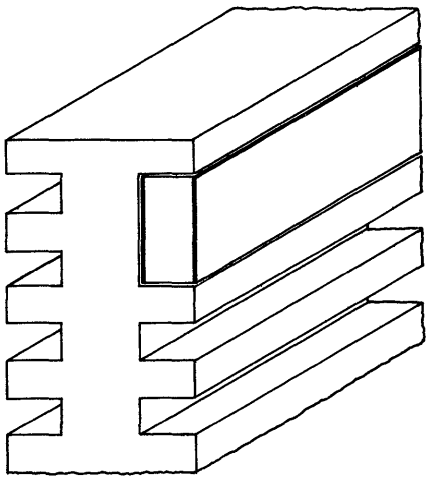
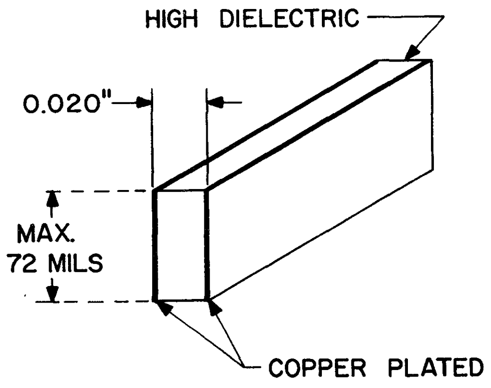



Figure 13 CERAMIC TRANSMISSION LINE IN THE CHANNEL WIRING ASSEMBLY

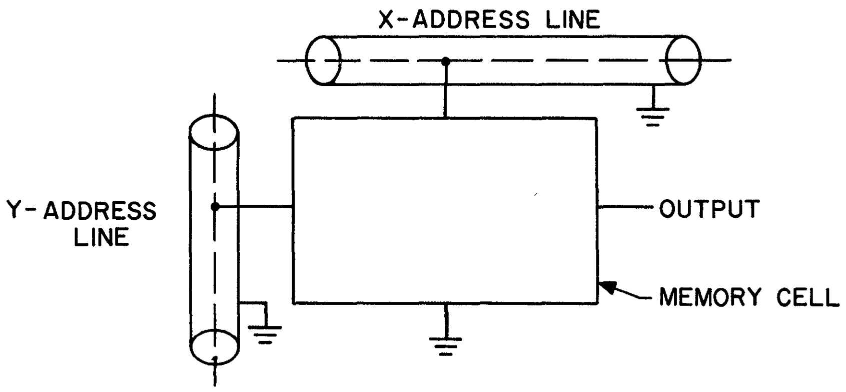


Figure 14 BIT ORGANIZED MEMORY CELL

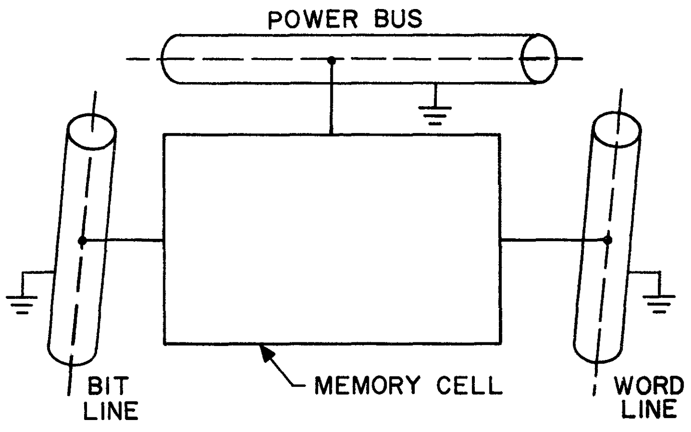


Figure 15 WORD ORGANIZED MEMORY CELL

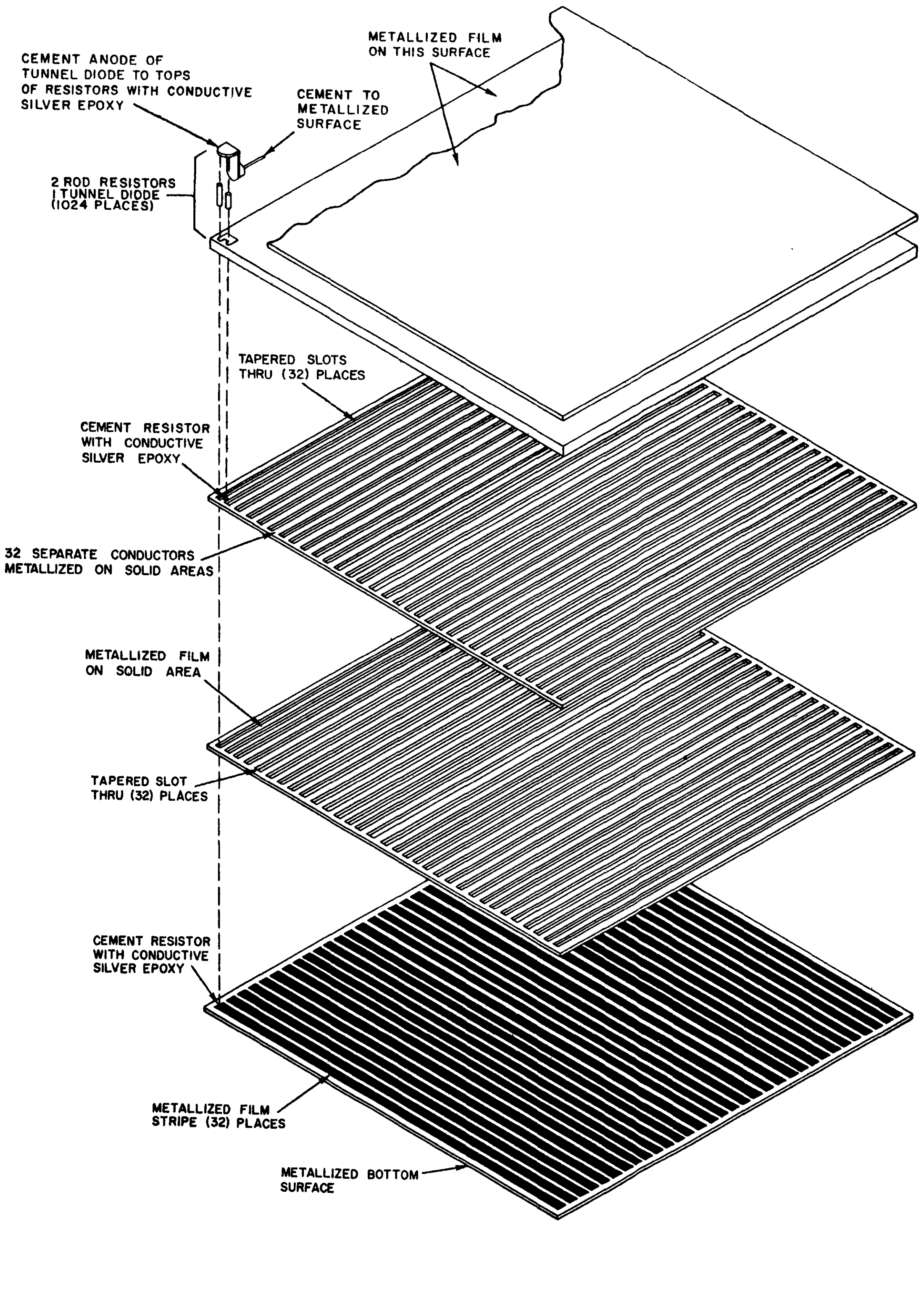

CEMENT ANODE OF TUNNEL DIODE TO TOPS OF RESISTORS WITH CONDUCTIVE SILVER EPOXY
METALLIZED FILM ON THIS SURFACE
CEMENT TO METALLIZED SURFACE
2 ROD RESISTORS 1 TUNNEL DIODE (1024 PLACES)
TAPERED SLOTS THRU (32) PLACES
CEMENT RESISTOR WITH CONDUCTIVE SILVER EPOXY
32 SEPARATE CONDUCTORS METALLIZED ON SOLID AREAS
METALLIZED FILM ON SOLID AREA
TAPERED SLOT THRU (32) PLACES
CEMENT RESISTOR WITH CONDUCTIVE SILVER EPOXY
METALLIZED FILM STRIPE (32) PLACES
METALLIZED BOTTOM SURFACE

Figure 16 EXPLODED VIEW OF MEMORY CONSTRUCTION FOR THE SINGLE TUNNEL DIODE MEMORY CIRCUIT

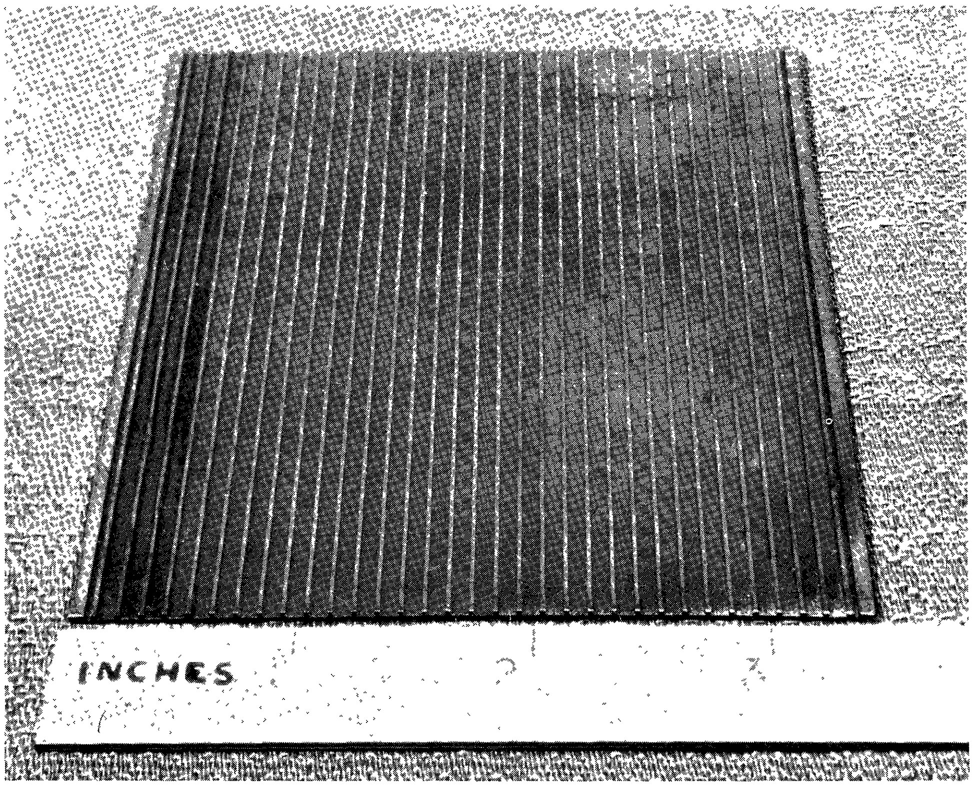


Figure 17. "Bit" Transmission Line of Memory Plane

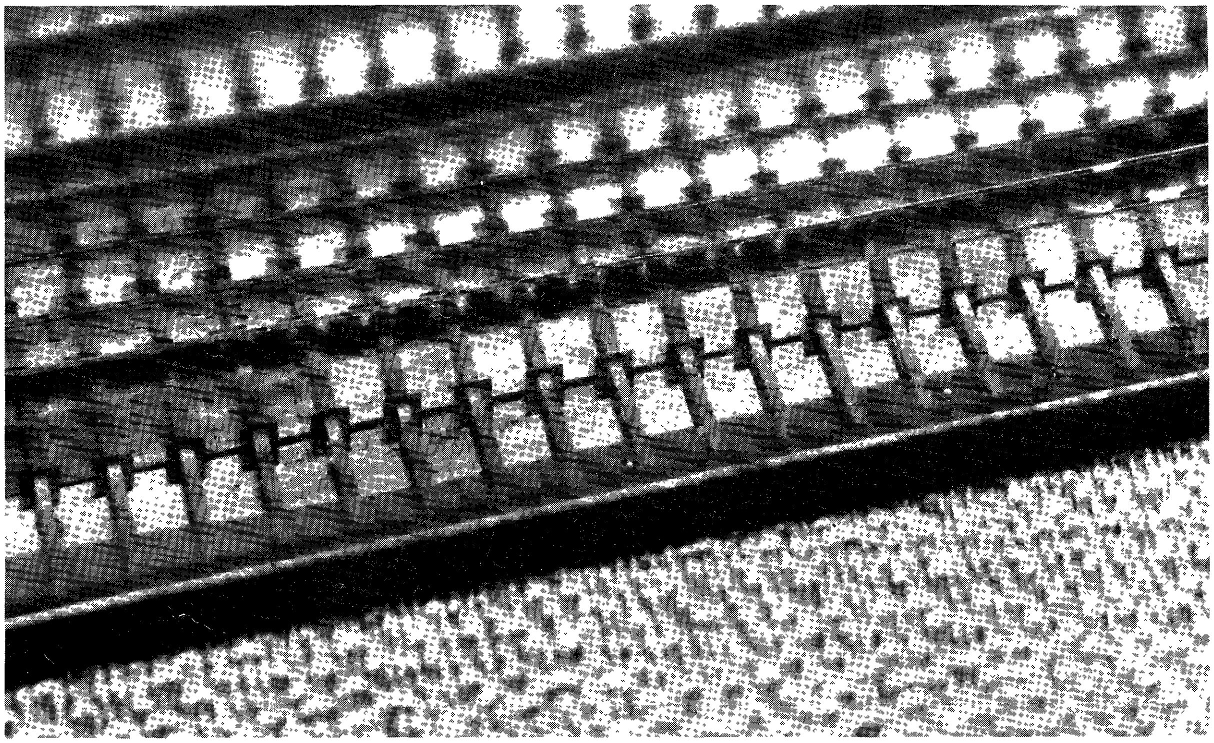


Figure 18. "Word" and "Supply Bus" Transmission
Lines Bridge the "Bit" Transmission Lines

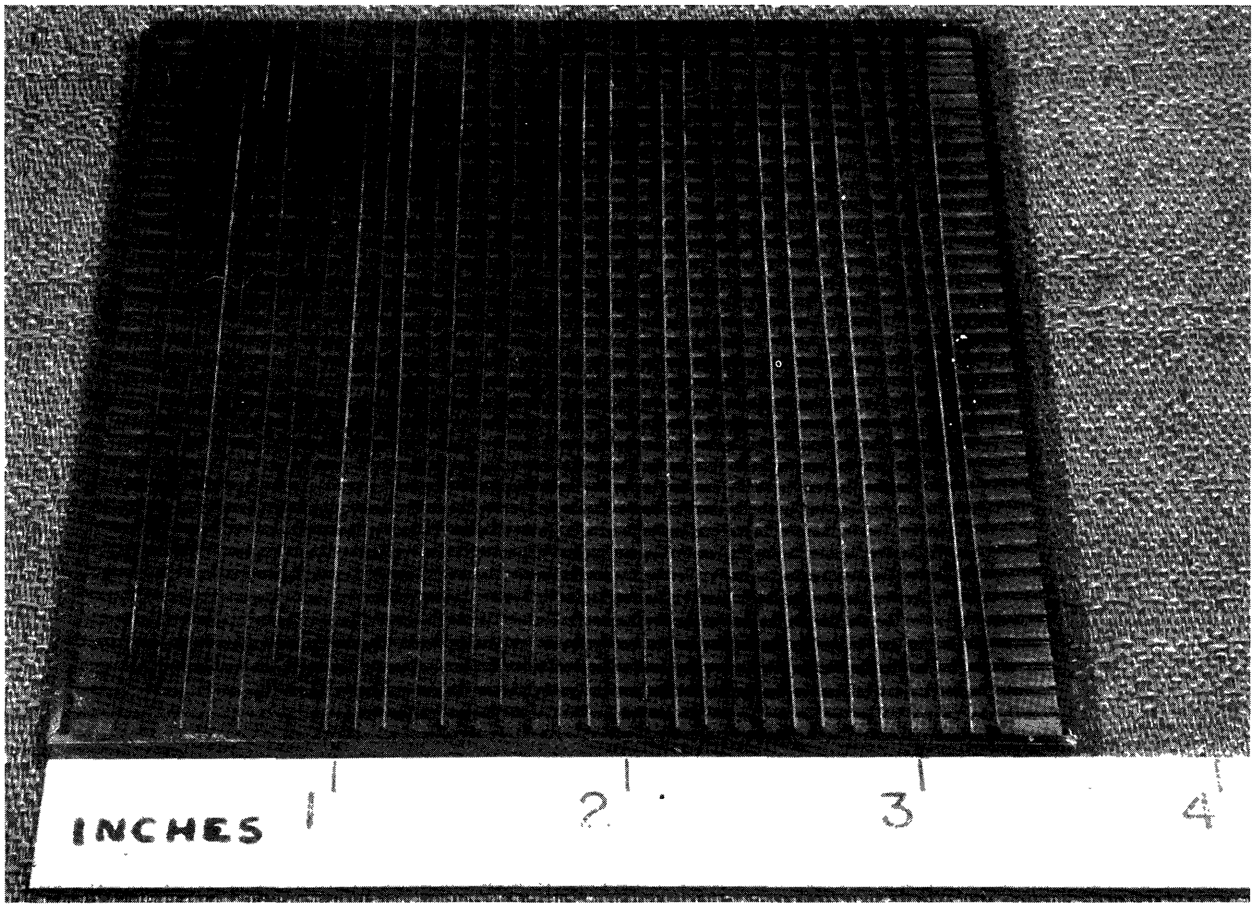


Figure 19 32 BY 32-BIT MEMORY PLANE WITH ITS THREE TRANSMISSION LINES

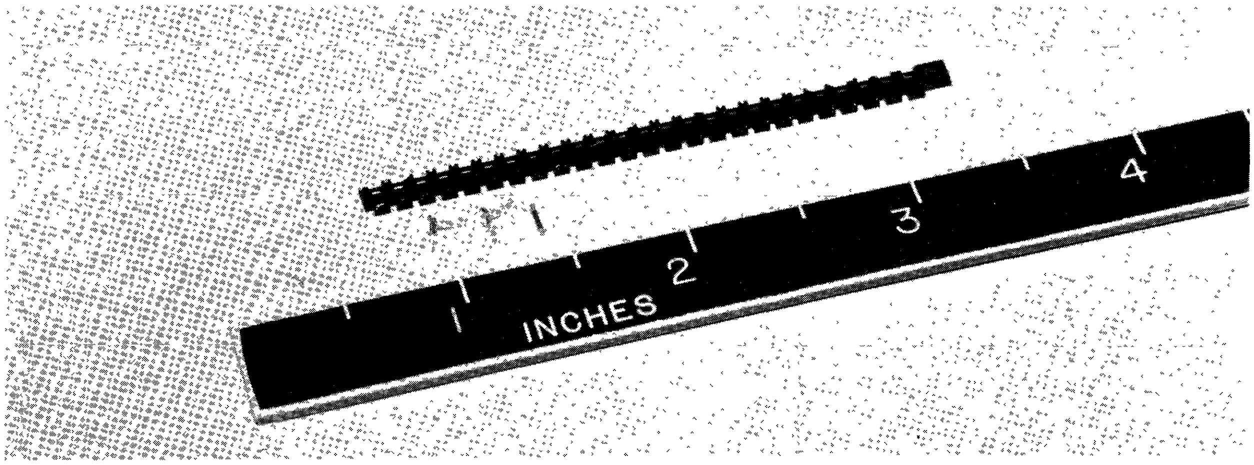


Figure 20 COMPONENTS SUBASSEMBLY FOR 32 BITS

MODELING HUMAN MENTAL PROCESSES

Herbert A. Simon
The RAND Corporation
Santa Monica, California

and

Carnegie Institute of Technology
Pittsburgh, Pennsylvania

There now exist at least a half dozen computer programs that simulate some of the information processes that humans use to perform problem solving, learning, perceiving, and thinking tasks. These programs constitute theoretical explanations of the corresponding human behavior, and can be tested by comparing the computer traces they produce with the verbal behavior of subjects in the psychological laboratory. This paper surveys this new kind of theory building and theory testing in psychology, and relates it to other uses of simulation as a tool of psychological research.

The use of computers to perform "humanoid" tasks--which provides the theme for this conference--falls into a number of distinguishable, though overlapping, categories. On the one hand, the goal may be to learn about human processes by simulating them; this has been the central motivation in simulating neural nets and a good part of the work on simulating human problem solving. On the other hand, the goal may be to find effective machine processes for accomplishing complex tasks--imitating the human processes only when this proves the most efficient way to do the job. This goal of "artificial intelligence" has been perhaps the primary motivation in the fields of information retrieval and language translation. The work to be described in this session falls in the former category: it is aimed at understanding the human mind by imitating it.

Some Kinds of Simulation of Mind

Computer simulations of human thinking can be classified along another dimension: the closeness of the simulation to, or its remoteness from, underlying physiological processes. We can distinguish at least the following broad categories:

1. Abstract simulation of adaptive, goal-seeking, learning mechanisms. Here the primary goal is to understand the nature of organisms in general, rather than the human organism in particular. One set of examples were the "tortoises" of Grey Walter,[1] mobile analogue computers that demonstrated "in the metal" that artifices can be constructed which will behave adaptively in an environment in response to drives, and will improve their adaptation through learning. Another example--also an analogue--is W. Ross Ashby's homeostat,[2] that shows how learning can be implemented through "Darwinian' mechanisms that cause mutations in the individual organism's program of adaptation to his environment.

2. Simulation of the sensory-perceptual processes by which humans recognize visual and aural patterns and symbols. Mechanical reception and decoding of human speech is a long-time goal of fundamental and applied research that has not yet reached complete success. But much is now known of the cues that humans use to recognize the basic phonemic units of spoken language, and within the past two years some partial successes have been achieved in mechanizing that recognition.[3,4] Even greater progress has been made with the simpler task of recognizing and decoding hand-sent Morse Code.[5,6] The classical pattern-recognition experiments of Selfridge and Dinneen[7] undertook to simulate some of the basic coding processes employed by the human retina.

3. Simulation of the self-organizing capabilities of neural nets. As in the work mentioned in the previous category, the problem that has usually been posed is to explain the phenomenon of pattern recognition: how the nervous system, given its known gross characteristics, can learn to classify, say, patterns of light that fall on the retina. The work falling in category 3 is concerned less with the rules by which patterns are classified, and more with the ways in which these rules are acquired by the nervous system. Farley and Clark[8,9] represented the nervous system as a network of individual elements--schematized neurons--connected in a more or less random fashion, subsequent appropriate organization being induced by learning. A similar scheme, developed independently by Rochester, Holland, Haibt, and Duda,[10] was aimed at testing the particular

hypotheses about neural organization that had been put forth by the psychologist Hebb.[11] Rosenblatt's Perceptrons[12] continue this general line of investigation.

4. Simulation of the symbol-manipulating or information processes employed in learning by rote, in attaining concepts, and in solving problems. This category is most closely related to category 2, but with less emphasis on perceptual processes involving the peripheral sense organs, more emphasis upon non-numerical processes, and more emphasis on the construction of a formal information-processing theory of human mental processes. The research that the other participants in this session will report falls in this category, but before I introduce it, I should like to say something about the methodology involved.

Non-Numerical Computation

Mathematics is the classical tool for formalizing theories, and arithmetic or numerical analysis the tool for testing theories by comparing them with data. Progress toward formal theory in psychology, and the behavioral sciences generally, has been much impeded by the difficulties that are encountered in finding mathematical formulations that capture the significant aspects of the phenomena under study. The difficulty lies not merely in the complexity of the relations among the phenomena; it is even more deeply rooted in the incorrigibly "qualitative" character of the raw data.

How shall we, for example, characterize the data from a laboratory study of human problem solving in order to make these data amenable to mathematical and numerical analysis? We can count the number of problems a subject solves in a given time, and assign scores to batteries of problems on the basis of such counts. We can tally numbers of errors of various kinds. But the numbers we obtain in these ways are pale shadows of the subject's actual behavior--particularly his verbal behavior if he thinks aloud while solving the problem. When we record such behavior, we get data like these:

> S. Well, one possibility right off the bat is when you have just a PvT like that [the problem expression] the last thing you might use is that rule 9. I can get everything down to a P and just add a vT. So that's one thing to keep in mind ... I don't know if that's possible; but I think it is because I see that expressions (2) and (4) are somewhat similar.

How do we build a mathematical model for such a verbal stream, or for the underlying thought processes that carry the stream along? How much of the process have we captured if we encode the verbal statements and make counts of the numbers of statements of one kind or another?

Psychologists have commonly retreated from one or both horns of the dilemma. Some have steeled themselves against accusations of "softness" from their fellow scientists, and have continued to deal with complex human behavior in all its qualitative, unmathematized, richness and vagueness. This strategy is most evident in clinical psychology, whose norms of clarity and testability are very far from the standards of the natural sciences. But the same characteristics appear, to a milder degree, in the work of psychologists--notably the Gestaltists and the so-called Wurzburg School--who have continued to deal with complex human thinking and problem-solving behavior.[13] From them we have had valuable insights, but little in the way of testable theory stated in operational terms.

Other psychologists have preserved formal rigor by retreating to simple dichotomous button-pushing choice situations, to the study of reaction times, or to maze experiments with rats. For human and animal experiments involving elementary tasks of these kinds, a considerable body of experimental technique and data and even some formal theory (e.g., stochastic learning theory[14]) has developed, but at the cost of leaving a very wide gap between the phenomena that have been treated and the kinds of complex human thinking behavior that we should like to be able to explain.

Computers now open up a third course of action that requires no compromise. We can continue to deal with complex verbal behavior, but use the computer to simulate it without first encoding it or forcing it into mathematical form. For computers, in addition to their arithmetic capabilities, have, of course, quite general capabilities for manipulating symbols: reading symbols, writing symbols, copying symbols, erasing symbols, comparing symbols for identity or difference, behaving conditionally on the outcomes of such comparisons.

The research we are considering in this session exploits the non-numerical symbol-manipulating capacities of computers. Its basic strategy is to use these capacities to formulate programs that simulate, step by step, the non-numerical symbol-manipulating processes that (if the hypothesis is correct) humans use when they memorize syllables, acquire new concepts, or solve problems. Such a program, once formulated, can be tested by comparing the stream of symbols it generates in a problem situation (the computer trace) with the stream of verbalizations of human subjects in the same problem situation in the psychological laboratory.

Information Processing Theories

The products of this kind of research are programs that purport to explain complex human activities in terms of organized systems of simple information processes--symbol-manipulating processes. In what sense do such programs "explain" the behavior? Clearly they say little about the underlying neurophysiological and biochemical processes that occur in the central and peripheral nervous systems. How can we have an explanation of the behavior without understanding those underlying processes?

Levels of Explanation

We explain phenomena by reducing them to other phenomena that seem to us, somehow, simpler and more orderly. How did Mendel, for example, explain the relative frequencies of his different kinds of peas in successive generations? He postulated (without any direct observational evidence) underlying dominant and recessive factors passed on from parents to their progeny, whose interaction determined the physical type of the progeny. Only many years later was any direct evidence obtained of microscopic structures in the cell--the chromosomes--that could provide the biological substrate for Mendel's "factors." Again, Morgan's studies of fruit fly populations led him to postulate even tinier components of the chromosomes--the genes. These had to await the electron microscope before they could be shown, by direct observation, to exist; and even today, we are still far from an explanation of these biological structures at the next, biochemical level.

The goal, then, in simulating complex human behavior is the same as the goal in simulating neural nets: We wish to explain the behavior. But the information processing theories approach that explanation in stages. They first reduce the complex behavior to symbol manipulating processes that have not, as yet, been observed directly in the human brain. The hope, of course, is that when we know enough about these processes, it will be possible to explain them at a still more fundamental level by reducing them to systems of neural events.

When this stage is reached, theories in psychology will begin to resemble theories in genetics and in the biophysical sciences in their hierarchical structure. At the highest (but least fundamental) level will be information processing theories of overt behavior. At the next level will be neurological theories explaining how elementary information processes are implemented in the brain. At a still more fundamental level will be biochemical theories reducing the neurological mechanisms to physical and chemical terms. Information processing theories of thinking, neurological theories, and biochemical theories are complementary, not competitive, scientific commodities. We shall need all three kinds, and perhaps others as well, before we shall understand the human mind.

Finally, when we use computers to state and test information processing theories of thinking, we do not postulate any crude analogy between computer and brain. We use the computer because it is capable of simulating the elementary information processes that these theories postulate as the bases for thinking. We do not assert that there is any resemblance between the electronic means that realize these processes in the computer and the neurological means that realize the corresponding processes in the brain. We do assert that, at a grosser level, the computer can be organized to imitate the brain.

Information Processing Languages

There has been a strong, and not accidental, interaction between work on the computer simulation of human thinking and research on computer programming. The kinds of processes that computers are called upon to perform when they are simulating thinking tend to be quite different from the processes they perform when they are carrying out numerical analyses. A superficial difference is that the former processes involve little or no use of arithmetic operations. A more fundamental difference is that memory must be organized in quite distinct ways in the two situations.

Within the past five years there have been a number of reports to these conferences on the general characteristics and specific structure of information processing languages specially designed to facilitate non-numerical simulation.[15,16] I shall not go over this familiar ground again, except to point out that when such languages are used to build psychological theories the languages themselves contain implicit postulates--although rather weak ones--about the way in which the central nervous system organizes its work.

One of the common characteristics of all of these languages is their organization of memory in lists and list structures. By this means there can be associated with any symbol in memory a "next" symbol--the symbol that follows it on the list to which they both belong. By the use of a slightly more complicated device, the description list, there can be associated with any symbol in memory a list of its attributes and their values. If the symbol, for example, represents an apple, we can store on its description list the fact that its color is red, its printed name is APPLE, and its spoken name, APUL. The incorporation of these two forms of association--the serial order of simple lists and the partial ordering of description lists--in information processing languages permits one to represent many of the associative properties of human memory in a quite simple and direct way. We can use simple lists to simulate serial memory--e.g., remembering the alphabet--and description lists to simulate paired associations--e.g., the association between an object as recognized visually and its name.

A characteristic of the list processing languages, which they share with most other compiling and interpretive languages, is that they organize behavior in hierarchical fashion. Routines use subroutines, which have their own subroutines, and so on. This characteristic of the languages again facilitates the construction of programs to simulate human behavior, which appears to be organized in a highly similar hierarchical manner. The fact that most investigators have found it easier to write simulation programs in interpretive list languages than in machine language derives, in all likelihood, from the fact that the former languages have already taken the first steps in the direction of organizing the computer processes to mirror the organization of the human mind.

Heuristic Problem Solving Programs

The Program of Selfridge and Dinneen

The work of Selfridge and Dinneen on pattern recognition,[7] which I earlier assigned to the second category of simulation programs--simulation of sensory-perceptual processes--really marks a transition to information processing simulations. The Selfridge-Dinneen program specified a set of processes to enable a computer to learn to discriminate among classes of patterns presented on a two-dimensional "retina." The patterns could represent, for example, English letters like "A" and "O" of varying shape, size, and orientation.

In the Selfridge-Dinneen program, recognition was accomplished by using various operators to transform the retinal stimuli--in general to simplify and "stylize" them--and then searching for characteristics of the transformed stimuli that grouped the various exemplars of a given alphabetic letter together, but separated the exemplars of different letters. Although the program made use of the arithmetic instructions of the computer, the operations were basically topological and non-numerical in nature. Appropriate organization rather than rapid arithmetic was at the heart of the program.

The Selfridge-Dinneen program foreshadowed subsequent work in this area in another important respect also. The characteristics used to distinguish patterns were heuristic. They amounted to rules of thumb, selected by the computer over a series of learning trials on the sole basis that they usually worked--that is, made the desired discriminations. In more traditional uses of computers it is usually required that the programs be algorithms--that they be systematic procedures which guarantee solution of the problem to a desired degree of accuracy. The heuristics generated by the pattern recognizing program provided no such guarantees. Since there are vast ranges of tasks, handled every day by human beings, for which no algorithms in the sense just indicated are known to exist, the admission of heuristics as program components opened the way to simulating the less systematic, but often effective, processes that characterize much garden-variety, everyday human thinking.

Subsequent work has tended to confirm this initial hunch, and to demonstrate that heuristics, or rules of thumb, form the integral core of human problem-solving processes. As we begin to understand the nature of the heuristics that people use in thinking, the mystery begins to dissolve from such (heretofore) vaguely understood processes as "intuition" and "judgment."

Some Other Problem-Solving Programs

In the period 1956 to 1958 there came into existence a number of other computer programs that accomplished complex tasks with a "humanoid" flavor: composing music,[17] playing checkers,[18] discovering proofs for theorems in logic,[19] and geometry,[20] designing electric motors and transformers,[21] playing chess,[22,23,24] and balancing an assembly line.[25] The primary goal in constructing most of these programs was to enable the computer to perform an interesting or significant task. Detailed simulation of the ways in which humans perform the same task was only a secondary objective--or was not considered at all.

Nevertheless, it was discovered that often the best program for doing the job was a program that incorporated some of the heuristics that humans used in doing such jobs. Thus, the music composition program of Hiller and Isaacson made use of some of the rules of classical counterpoint; the motor design programs and line balancing program were generally organized in much the same ways as the procedures of experienced engineers, and so on. Hence, to a greater or lesser degree, all of these programs have taught us something about the ways in which people handle such tasks--especially about some of the kinds of heuristics they use.

Among these programs Samuel's checker program and the Los Alamos chess program place the least emphasis on heuristics, and hence provide valuable yardsticks for comparison with heuristic programs handling the same, or similar tasks. These two programs make essential use of the computer's capabilities for extremely rapid arithmetic, for their basic strategy is to look at all possible (legal) continuations of the game for several moves ahead, and then to choose that move which appears most favorable (in a minimax sense) in terms of the possible outcomes. In contrast, Bernstein's and the NSS chess programs examine a small, highly selective subset of all possible continuations of the game and choose a move that appears good in the light of this selective analysis.

Thus, the Los Alamos program, looking two moves ahead, will typically examine a little less than a million possible continuations, Bernstein's program approximately 2,500, and the NSS program almost never more than one hundred and more usually only a handful. All three programs play roughly the same quality of chess (mediocre) with roughly the same amount of computing time. The effort saved by the heuristic programs in looking at fewer continuations, is expended in selecting more carefully those to be examined and subjecting them to more thorough examination. Thus, the more systematic, arithmetic programs provide benchmarks against which the progress in developing heuristics can be measured.

The General Problem Solver

All of the programs we have mentioned fell short of human simulation in one very fundamental respect--apart from failures of detail. They were all special-purpose programs. They enabled the computer to perform one kind of complex task, and one kind only. Only in a few cases (the Checker Player[18] and the Logic Theorist[26]) did they enable the computer to improve its performance through learning. Yet we know that the human mind is (a) a general-purpose mechanism and (b) a learning mechanism. A person who is brought into a relatively novel task situation may not handle the situation with skill but, unless it is inordinately difficult, will not find himself at a complete loss. Whether he succeeds in solving the problem that is posed him, or not, he is able, at least, to <u>think</u> about it.

We must conclude that if a computer program is to simulate the program that a human brings to a problem situation, it must contain two components: (a) a general-purpose thinking and learning program that makes no direct reference to any particular task or subject matter; and (b) heuristics that embody the specific techniques and procedures which make possible the skilled and efficient performance of particular classes of tasks. The program must incorporate both general intelligence and special skills.

The General Problem Solver (GPS) was the first computer program aimed at describing the problem solving techniques used by humans that are independent of the subject matter of the problem.[27]

Since GPS has been described elsewhere, I shall say only a word about its structure. It is a program for achieving the goal of transforming a particular symbolic object (representing the "given" problem situation) into a different symbolic object (the "desired" situation or goal situation). It does this by discovering differences between pairs of objects, and by searching for operators that are relevant to reducing these differences. In the form in which it has thus far been realized on a computer, GPS is not a learning program, hence still falls far short of simulating all aspects of what we would call general intelligence.

In its current computer realization, GPS has solved some simple problems of finding proofs for theorems in symbolic logic (substantially the same task as that handled by the special-purpose Logic Theorist). It has solved the well-known puzzle of Missionaries and Cannibals--finding a plan for transporting three missionaries and three cannibals across a river without any of the missionaries being eaten. Hand simulation has demonstrated that it can handle trigonometric and algebraic identities. On the basis of other investigations that have not fully reached the programming stage, it appears highly likely that GPS will be able to solve certain tactical problems in chess (e.g., to find a move leading to a fork of a pair of enemy pieces), do formal differentiation and integration, and write codes for simple computer programs in IPL V. Several possibilities for incorporating learning processes in GPS, one of them using GPS in the learning mechanism itself,[28] have also been explored.

The adequacy of GPS as a simulation of human problem solving has been examined, primarily in the task domain of symbolic logic, by comparing the computer trace with the thinking-aloud protocols of college students solving identical problems.[29] The evidence to date suggests that GPS does indeed capture the principal problem-solving methods used by the human subjects. The detailed comparison of its behavior with the protocols has cast considerable light on the processes of abstraction and on the nature and uses of imagery in problem solving.

Recent Advances in the Simulation of Thinking

The remaining papers to be presented in this session will describe a number of heuristic programs that have been written in the past two years, and which extend very substantially the range of human mental processes that have been simulated with these techniques. I shall not anticipate the content of these programs, beyond indicating what their relation is to those I have already mentioned.

Areas of Psychological Experimentation

The simulations mentioned so far all fall in the area that psychologists call "higher mental processes." As I indicated earlier, these processes have tended to be underemphasized in American experimental psychology until quite recently because we did not have tools for investigating them in an objective and rigorous way. If computer simulation has shown itself to be a powerful tool of research in an area as difficult as the study of higher mental processes, we might expect this tool to prove even more powerful if applied to the simpler phenomena with which experimental psychologists have been largely concerned. The papers of this session report some of the first evidence that this expectation is justified.

What are the kinds of tasks and processes that have been most thoroughly studied by psychologists? Perception--the interaction of sensory organs and central nervous system in the discrimination and recognition of stimuli--has been the subject of extensive investigation. A second, very active, research area has been learning, and particularly the rote learning of serial material and of stimulus-response pairs. A third area has been simple choice behavior, especially choice among a small number (usually two) of alternatives with systematic or intermittent reward. Animal and human maze learning experiments have been used to study both rote learning and simple choice behavior. Finally, there is a rather varied assortment of work that is usually classified under the heading of "concept formation" or "concept attainment."

No one supposes that the topics I have mentioned--perception, rote learning, simple choice behavior, maze learning, and concept formation--are mutually exclusive and exhaustive categories. They are simply pigeon holes that psychologists have found convenient for classifying experiments. It is almost certain that the mechanisms required to perform tasks in one of these areas are called into play in some of the others. Hence, we would have reason to hope that as heuristic programs are constructed to handle one or another of these tasks, the mechanisms employed in the several programs will begin to show distinct resemblances--and resemblances also to the mechanisms used in problem-solving simulations. Such resemblances and common mechanisms are already beginning to appear.

Long-Range Goals of Simulation

The long-term research strategy would again be gradually to replace a multitude of special-purpose programs with a more general program aimed at simulating the whole man--or at least the cognitive aspects of his behavior. Although enormous gaps of ignorance still separate us from that goal, the goal itself no longer seems entirely Utopian to the active researchers in the field.

Perhaps the largest single gap at present--and one that is not filled by any of the work to be reported today--is in programs to explain long-range human memory phenomena. I will venture the personal prediction that filling this gap will soon become crucial to progress in the whole field of information retrieval.

Another important gap that also has significant practical implications lies in the area of simulation of natural language processes. Here, interest in language translation and in the improvement of computer programming languages has already led to exciting progress--as illustrated, for example, by the work of Chomsky[30] and Yngve.[31]

Heuristic Programs in New Areas

The areas of rote learning, simple choice behavior, and concept attainment are represented in the programs to be described by Mssrs. Feigenbaum, Feldman, and Hunt, respectively.

Rote Learning. The Elementary Perceiver and Memorizer (EPAM) is a theory to explain how human subjects store in memory symbolic materials that are inherently "meaningless." The typical learning materials are "nonsense syllables"--spoken or printed syllables that do not correspond to English words. By studying rote learning, we hope to understand, for example, how humans learn to associate names with objects, and learn to read by associating printed words with their oral counterparts.

Binary Choice. In the so-called partial reinforcement or binary choice experiment, the subject is instructed to guess which of two events will occur next. In variants of the experiment, the actual event sequence may be patterned, or it may be a random sequence. The binary choice experiment has been one of the principal situations used to test the stochastic learning models that have been developed in psychology over the last decade.[14,32] Mr. Feldman's Binary Choice program offers an alternative theory to explain these phenomena, hence provides an interesting example for comparing and contrasting heuristic programs with more traditional mathematical models.

Concept Formation. In the simplest form of the concept formation task, a rat is given a choice of two gates, one of which is labelled, say, with a large triangle, the other with a small circle. If the experimenter's aim is to test the rat's attainment of the concept "triangle," he places a reward behind the gate labelled with the triangle. On succeeding trials, the symbols change in shape, size, or color, but the gate labelled with a triangle always leads to the reward. Within the past year, several computer programs have been written that simulate slightly more complex concept learning behavior in humans. One of these programs, the Concept Learner, will be described by Hovland and Hunt.

Conclusion

I have tried to outline the development over the past decade of the use of computers to construct and test non-numerical information-processing explanations for human thinking and learning. Such programs, which are beginning to be validated by behavioral evidence, are providing embryonic theories for these phenomena in terms of underlying information processes. Hopefully, the elementary information processes that are postulated in the theories will, in turn, find their explanation in neurological processes and mechanisms. The papers in this session describe a few of the programs of this kind that have been constructed to date, and provide some basis for judging the prospects for this approach to understanding the human mind.

References

1. Walter, Grey, The Living Brain, Van Nostrand, 1953.

2. Ashby, W. R., Design for a Brain, Wiley, 1952.

3. Fatehchand, R., Machine recognition of spoken words, in F. L. Alt, ed., Advances in Computers, Academic Press, 1960, pp. 193-231.

4. Forgie, J. W. and C. D. Forgie, 'Results Obtained from a Vowel Recognition Computer Program," Jour. of the Acoustical Society of America, 31:1480-1489 (November, 1959).

5. Gold, B., 'Machine Recognition of Hand-Sent Morse Code," IRE Trans. on Information Theory, IT-5; #1:17-24 (March, 1959).

6. Blair, C. R., "On Computer Transcription of Manual Morse,' Jour. of the Association for Computing Machinery, 6:429-442 (July, 1959).

7. Selfridge, O.G., "Pattern Recognition and Modern Computers' and Dinneen, G.P., "Programming Pattern Recognition," Proc. of the 1955 Western Joint Computer Conference, pp. 91-100.

8. Farley, B. G. and W. A. Clark, 'Simulation of Self-organizing Systems by Digital Computer," IRE Trans. of the Professional Group on Information Theory, PGIT-4: 76-84 (September, 1954).

9. Clark, W. A. and B. G. Farley, 'Generalization of Pattern Recognition in a Self-organizing System,' Proc. of the 1955 Western Joint Computer Conference, pp. 86-91.

10. Rochester, N., J. H. Holland, L. H. Haibt and W. L. Duda, Tests on a Cell Assembly Theory of the Action of the Brain Using a Large Digital Computer,' IRE Trans. on Information Theory, IT-2; #3, (September, 1956).

11. Hebb, D. O., Organization of Behavior, Wiley, 1949.

12. Rosenblatt, F., "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain,' Psychological Review, 65: 386-408 (November, 1958).

13. Johnson, D. M., The Psychology of Thought and Judgment, Harper, 1955.

14. Bush, R. R. and F. Mosteller, Stochastic Models for Learning, Wiley, 1955.

15. Newell, A. and J. C. Shaw, "Programming the Logic Theory Machine," Proceedings of the 1957 Western Joint Computer Conference, pp. 230-240.

16. Shaw, J. C., A. Newell, H. A. Simon, and T. O. Ellis, "A Command Structure for Complex Information Processing," Proc. of the 1958 Western Joint Computer Conference, pp. 119-128.

17. Hiller, L. A. and L. M. Isaacson, Experimental Music, McGraw-Hill, 1959.

18. Samuel, A. L., "Some Studies in Machine Learning using the Game of Checkers," IBM J. of Research and Development, 3: 210-229 (July, 1959).

19. Newell, A. and H. A. Simon, 'The Logic Theory Machine," IRE Trans. on Information Theory, IT-2; #3: pp. 61-79, (September, 1956).

20. Gelernter, H. and N. Rochester, "Intelligent Behavior in Problem-Solving Machines," IBM J. of Research and Development, 2:336-345 (October, 1958).

21. Goodwin, G. L., 'Digital Computers Tap Out Designs for Large Motors Fast,' Power, April, 1958.

22. Kister, J., P. Stein, S. Ulam, W. Walden and M. Wells, 'Experiments in Chess,' Jour. of the Association for Computing Machinery, 4:174-177 (April, 1957).

23. Bernstein, A., M. de V. Roberts, T. Arbuckle and M. H. Belsky, "A Chess-Playing Program for the IBM 704," Proc. of the 1958 Western Joint Computer Conference, pp. 157-159.

24. Newell, A., J. C. Shaw and H. A. Simon, "Chess Playing Programs and the Problem of Complexity," IBM J. of Research and Development, 2:320-335 (October, 1958).

25. Tonge, F. M., A Heuristic Program for an Assembly Line Balancing Problem, Prentice-Hall, forthcoming.

26. Newell, A., J. C. Shaw and H. A. Simon, "Empirical Explorations of the Logic Theory Machine," Proc. of the 1957 Western Joint Computer Conference, pp. 218-230.

27. Newell, A., J. C. Shaw and H. A. Simon, "A General Problem-Solving Program for a Computer," Computers and Automation, 8:10-17 (July, 1959).

28. Newell, A., J. C. Shaw and H. A. Simon, "A Variety of Intelligent Learning in a General Problem Solver," in M. C. Yovits and S. Cameron, eds., Self-Organizing Systems, Pergamon, 1960, pp. 153-189.

29. Newell, A. and H. A. Simon, "The Simulation of Human Thinking," in Current Trends in Psychology, 1959, U. of Pittsburgh Press, 1960.

30. Chomsky, A. N., Syntactic Structures, Mouton, 1957.

31. Yngve, V., "A Model and an Hypothesis for Language Structure," Proc. of the American Philosophical Society, 104:444-466, October, 1960.

32. Suppes, P. and R. C. Atkinson, Markov Learning Models for Multiperson Interactions, Stanford U. Press, 1960.

THE SIMULATION OF VERBAL LEARNING BEHAVIOR*

E. A. Feigenbaum
University of California
Berkeley, California

and

The RAND Corporation
Santa Monica, California

Summary

An information processing model of elementary human symbolic learning is given a precise statement as a computer program, called Elementary Perceiver and Memorizer (EPAM). The program simulates the behavior of subjects in experiments involving the rote learning of nonsense syllables. A discrimination net which grows is the basis of EPAM's associative memory. Fundamental information processes include processes for discrimination, discrimination learning, memorization, association using cues, and response retrieval with cues. Many well-known phenomena of rote learning are to be found in EPAM's experimental behavior, including some rather complex forgetting phenomena. EPAM is programmed in Information Processing Language V.

H. A. Simon has described some current research in the simulation of human higher mental processes and has discussed some of the techniques and problems which have emerged from this research. The purpose of this paper is to place these general issues in the context of a particular problem by describing in detail a simulation of elementary human symbolic learning processes.

The information processing model of mental functions employed is realized by a computer program called Elementary Perceiver and Memorizer (EPAM). The EPAM program is the precise statement of an information processing theory of verbal learning that provides an alternative to other verbal learning theories which have been proposed.** It is the result of an attempt to state quite precisely a parsimonious and plausible mechanism sufficient to account for the rote learning of nonsense syllables. The critical evaluation of EPAM must ultimately depend not upon the interest which it may have as a learning machine, but upon its ability to explain and predict the phenomena of verbal learning.

I should like to preface my discussion of the simulation of verbal learning with some brief remarks about the class of information processing models of which EPAM is a member.

a. These are models of mental processes, not brain hardware. They are psychological models of mental function. No physiological or neurological assumptions are made, nor is any attempt made to explain information processes in terms of more elementary neural processes.

b. These models conceive of the brain as an information processor with sense organs as input channels, effector organs as output devices, and with internal programs for testing, comparing, analyzing, rearranging, and storing information.

c. The central processing mechanism is assumed to be serial; i.e., capable of doing only one (or a very few) things at a time.

d. These models use as a basic unit the information symbol; i.e., a pattern of bits which is assumed to be the brain's internal representation of environmental data.

e. These models are essentially deterministic, not probabilistic. Random variables play no fundamental role in them.

*I am deeply indebted to Herbert A. Simon for his past and present collaboration in this research. This research has been supported by the Computer Sciences Department, The RAND Corporation, and the Ford Foundation. I wish to express appreciation for the help and critical comments of Julian Feldman, Allen Newell, J. C. Shaw and Fred Tonge.

**Examples of quantitative (or quasi-quantitative) theories of verbal learning are those of Hull, et.al. [1], Gibson [2], and Atkinson [3].

THE BASIC EXPERIMENT

Early in the history of psychology, the psychologist invented an experiment to simplify the study of human verbal learning. This "simple" experiment is the rote memorization of nonsense syllables in associate-pairs or serial lists.

The items to be memorized are generally three-letter words having consonant letters on each end and a vowel in the middle. Nonsense syllables are chosen in such a way that the three-letter combinations have no ordinary English meaning. For example, CAT is not a nonsense syllable, but XUM is.*

In one basic variation, the rote memory experiment is performed as follows:

a. A set of nonsense syllables is chosen and the syllables are paired, making, let us say, 12 pairs.

b. A subject is seated in front of a viewing apparatus and the syllables are shown to him, one pair at a time.

c. First, the left-hand member of the pair (stimulus item) is shown. The subject tries to say the second member of the pair (response item).

d. After a short interval, the response item is exposed so that both stimulus and response items are simultaneously in view.

e. After a few seconds, the cycle repeats itself with a new pair of syllables. This continues until all pairs have been presented (a trial).

f. Trials are repeated, usually until the subject is able to give the correct response to each stimulus. There is a relatively short time interval between trials.

g. For successive trials the syllables are reordered randomly. This style of carrying out the experiment is called paired-associates presentation.

The other basic variant of the experiment is called serial-anticipation presentation. The nonsense syllables (say, 10 or 12 items) are arranged in a serial list, the order of which is not changed on successive trials. When he is shown the nth syllable, the subject is to respond with the (n+1)st syllable. A few seconds later, the (n+1)st syllable is shown and the subject is to respond with the (n+2)nd syllable, and so on. The experiment terminates when the subject is able to correctly anticipate all of the syllables.

Numerous variations on this experimental theme have been performed.* The phenomena of rote learning are well studied, stable, and reproducible. For example, in the typical behavioral output of a subject, one finds:

a. Failures to respond to a stimulus are more numerous than overt errors.

b. Overt errors are generally attributable to confusion by the subject between similar stimuli or similar responses.

c. Associations which are given correctly over a number of trials sometimes are then forgotten, only to reappear and later dissappear again. This phenomenon has been called oscillation.**

d. If a list x of syllables or syllable pairs is learned to the criterion; then a list y is similarly learned; and finally retention of list x is tested; the subject's ability to give the correct x responses is degraded by the interpolated learning. The degradation is called retroactive inhibition. The overt errors made in the

*People will defy an experimenter's most rigorous attempt to keep the nonsense syllables association-free. Lists of nonsense syllables have been prepared, ordering syllables on the basis of their so-called "association value," in order to permit the experimenter to control "meaningfulness."

*For an extended treatment of this subject, see Hovland, C. I., "Human Learning and Retention." [4]

**By Hull [5]. Actually he called it "oscillation at the threshold of recall," reflecting his theoretical point of view.

retest trial are generally intrusions from the list y. The phenomenon disappears rapidly. Usually after the first retest trial, list x has been relearned back to criterion.

e. As one makes the stimulus syllables more and more similar, learning takes more trials.

The Information Processing Model

This section describes the processes and structures of EPAM.

EPAM is not a model for a particular subject. In this respect it is to be contrasted with the binary choice models of particular subjects which Mr. Feldman is presenting in this session. The fact is that individual differences play only a small part in the results of the basic experiment described above.

It is asserted that there are certain elementary information processes which an individual must perform if he is to discriminate, memorize and associate verbal items, and that these information processes participate in all the cognitive activity of all individuals.*

It is clear that EPAM does not yet embody a complete set of such processes. It is equally clear that the processes EPAM has now are essential and basic.

*Some information processing models are conceived as models of the mental function of particular subjects; e.g., Feldman's Binary Choice Model [6]. Others treat the general subject as EPAM does. Still others are mixed in conception, asserting that certain of the processes of the model are common for all subjects while other processes may vary from subject to subject; e.g., the General Problem Solver of Newell, Shaw and Simon [7]. Alternatively, information processing models may also be categorized according to how much of the processing is "hard core" (i.e., necessary and invariant) as opposed to "strategic" (i.e, the result of strategy choice by control processes). I suggest the obvious: that models of strategies for information processing will tend to be models of the general subject. As exemplars, Lindsay's Reading Machine [8], a "hard core" model, treats the general subject; Wickelgren's model of the conservative Focusing Strategy in concept attainment (Wickelgren [9]; Bruner, Goodnow, and Austin [10]), a pure strategy model, can predict only the behavior of particular subjects.

Overview: Performance and Learning

Conceptually, EPAM can be broken down into two subsystems, a performance system and a learning system. In the performance mode, EPAM produces responses to stimulus items. In the learning mode, EPAM learns to discriminate and associate items.

The performance system is the simpler of the two. It is sketched in Fig. 1. When a stimulus is noticed, a perceptual process encodes it, producing an internal representation (an input code). A discriminator sorts the input code in a discrimination net (a tree of tests and branches) to find a stored image of the stimulus. A response cue associated with the image is found, and fed to the discriminator. The discriminator sorts the cue in the net and finds the response image, the stored form of the response. The response image is then decoded by a response generator letter by letter in another discrimination net into a form suitable for output. The response is then produced as output.

The processes of the learning system are more complex. The discrimination learning process builds discriminations by growing the net of tests and branches. The association process builds associations between images by storing response cues with stimulus images. These processes will be described fully in due course.

The succeeding sections on the information processing model give a detailed description of the processes and structures of both systems.

Input to EPAM: Internal Representations of External Data

The following are the assumptions about the symbolic input process when a nonsense syllable is presented to the learner. A perceptual system receives the raw external information and codes it into internal symbols. These internal symbols contain descriptive information about features of external stimulus. For unfamiliar 3-letter nonsense symbols, it is assumed that the coding is done in terms of the individual letters, for these letters are familiar and are well-learned units for the adult subject.*

*The basic perception mechanism I have in mind is much the same as that of Selfridge [11] and Dinneen, whose computer program scanned letters and perceived simple topological features of these letters.

The end result of the perception process is an internal representation of the nonsense syllable--a list of internal symbols (i.e., a list of lists of bits) containing descriptive information about the letters of the nonsense syllable. Using Minsky's terminology [12], this is the "character" of the nonsense syllable.

I have not actually programmed this perception process. For purposes of this simulation, I have assigned coded representations for the various letters of the alphabet based on 15 different geometrical features of letters. For purposes of exploring and testing the model, at present all that is really needed of the input codes is:

a. that the dimensions of a letter code be related in some reasonable way to features of real letters.

b. that the letter codes be highly redundant, that is, include many more dimensions than is necessary to discriminate the letters of the alphabet.

To summarize, the internal representation of a nonsense syllable is a list of lists of bits, each sublist of bits being a highly redundant code for a letter of the syllable.

Given a sequence of such inputs, the essence of the learner's problem is twofold: first, to _discriminate_ each code from the others already learned, so that differential response can be made; second, to _associate_ information about a "response" syllable with the information about a "stimulus" syllable so that the response can be retrieved if the stimulus is presented.

Discriminating and Memorizing: Growing Trees of Images

I shall deal with structure first and reserve my discussion of process for a moment.

Discrimination net. The primary information structure in EPAM is the _discrimination net_. It embodies in its _structure at any_ moment all of the discrimination learning that has taken place up to a given time. As an information structure it is no more than a familiar friend: a sorting tree or decoding network. Fig. 2 shows a small net. At the terminals of the net are lists called _image lists_, in which symbolic _information can_ be stored. At the nodes of the net are stored programs, called _tests_, which examine characteristics _of an_ input code and signal branch-left or branch-right. On each image list will be found a list of symbols called the _image_. An image is a partial or total _copy of_ an input code. I shall use these names in the following description of net processes.

Net Interpreter. The discrimination net is examined and altered by a number of processes, most important of which is the _net interpreter_. The net interpreter _sorts an input code_ in the net and produces the image list associated with that input code. _This retrieval process is the essence of a purely associative memory: the stimulus information itself leads to the retrieval of the information associated with that stimulus. The net interpreter is a very simple_ process. It finds the test in the topmost node of the tree and executes this program. The resulting signal tells it to branch left or branch right to find the succeeding test. It executes this, tests its branches again, and repeats the cycle until a terminal is found. The name of the image list is produced, and the process terminates. This is the discriminator of the performance system which sorts items in a static net.

Discrimination Learning. The discrimination learning process of the learning system grows the net. Initially we give the learning system no discrimination net but only a set of simple processes for growing nets and storing new images at the terminals.

To understand how the discrimination and memorization processes work, let us examine in detail a concrete example from the learning of nonsense syllables. Suppose that the first stimulus-response associate-pair on a list has been learned. (Ignore for the moment the question of how the association link is actually formed.) Suppose that the first syllable pair was DAX-JIR. The discrimination net at this point has the simple two-branch structure shown in Fig. 3. Because the syllables differ in their first letter, Test 1 will probably be a test of some characteristic on which the letters D and J differ. No more tests are necessary at this point.

Notice that the image of JIR which is stored is a full image. Full response images must be stored--to provide the information for _producing_ the response; but only partial _stimulus_ images need be stored--to provide the information for _recognizing_ the stimulus. How much _stimulus image_ information is required the learning system determines for itself as it grows its discrimination net, and makes errors which it diagnoses as inadequate discrimination.

To pursue our simple example, suppose that the next syllable pair to be learned is PIB-JUK. There are no storage terminals in the net, as it stands, for the two new items. In other words, the net does not have the discriminative capability to contain more than two items. The input code for PIB is sorted by the net interpreter. Assume that Test 1 sorts it down the plus branch of Fig. 3. As there are differences between the incumbent image (with first-letter D) and the new code (with first-letter P) an attempt to store an image of PIB at this terminal would destroy the information previously stored there.

Clearly what is needed is the ability to discriminate further. A match for differences between the incumbent image and the challenging code is performed. When a difference is found, a new test is created to discriminate upon this difference. The new test is placed in the net at the point of failure to discriminate, an image of the new item is created, and both images--incumbent and new--are stored in terminals along their appropriate branches of the new test, and the conflict is resolved.*

*With the processes just described, the discrimination net would be grown each time a new item was to be added to the memory. But from an information processing standpoint, the matching and net-growing processes are the most time-consuming in the system. In general, with little additional effort, more than one difference can be detected, and more than one discriminating test can be added to the net. Each redundant test placed in the net gives one "empty" image list. At some future time, if an item is sorted to this empty image list, an image can be stored without growing the net. There is a happy medium between small nets which must be grown all the time and large nets replete with redundant tests and a wasteful surplus of empty image lists. Experimentation with this "structural parameter" has been done and it has been found that for this study one or two redundant tests per growth represents the happy medium. However, I would not care to speak of the generality of this particular result.

The net as it now stands is shown in Fig. 4. Test 2 is seen to discriminate on some difference between the letters P and D.

The input code for JUK is now sorted by the net interpreter. Since Test 1 cannot detect the difference between the input codes for JUK and JIR (under our previous assumption), JUK is sorted to the terminal containing the image of JIR. The match for differences takes place. Of course, there are no first-letter differences. But there are differences between the incumbent image and the new code in the second and third letters.

Noticing Order. In which letter should the matching process next scan for differences? In a serial machine like EPAM, this scanning must take place in some order. This order need not be arbitrarily determined and fixed. It can be made variable and adaptive. To this end EPAM has a <u>noticing order for letters of syllables</u>, which prescribes at any moment a letter-scanning sequence for the matching process. Because it is observed that subjects generally consider end-letters before middle-letters, the noticing order is initialized as follows: first-letter, third-letter, second-letter. When a particular letter being scanned yields a difference, this letter is promoted up one position on the noticing order. Hence, letter positions relatively rich in differences quickly get priority in the scanning. In our example, because no first-letter differences were found between the image of JIR and code for JUK, the third letters are scanned and a difference is found (between R and K). A test is created to capitalize on this third-letter difference and the net is grown as before. The result is shown in Fig. 5. The noticing order is updated; third-letter, promoted up one, is at the head.

Learning of subsequent items proceeds in the same way, and we shall not pursue the example further.

<u>Associating Images: Retrieval Using Cues</u>

The discrimination net and its interpreter associate codes of external objects with internal image lists and images. But the basic rote learning experiment requires that stimulus information somehow lead to response information and a response. The discrimination net concept can be used for the association of internal images with each other (i.e., response with stimulus) with very little addition to the basic mechanism.

An association between a stimulus image and a response image is accomplished by storing with the stimulus image some of the coded information about the response. This information is called the cue. A cue is of the same form as an input code, but generally contains far less information than an input code. A cue to an associated image can be stored in the discrimination net by the net interpreter to retrieve the associated image. If, for example, in the net of Fig. 3 we had stored with the stimulus image the letter J as a cue to the response JIR, then sorting this cue would have correctly retrieved the response image. An EPAM internal association is built by storing with the stimulus image information sufficient to retrieve the response image from the net at the moment of association.

The association process determines how much information is sufficient by trial and error. The noticing order for letters is consulted, and the first-priority letter is added to the cue. The cue is then sorted by the net interpreter and a response image is produced. It might be the wrong response image; for if a test seeks information which the cue does not contain, the interpreter branches left or right randomly (with equal probabilities) at this test.* During association, the selection of the wrong response is immediately detectable (by a matching process) because the response input code is available. The next-priority letter is added to the cue and the process repeats until the correct response image is retrieved. The association is then considered complete.

Note two important possibilities. First, by the process just described, a cue which is really not adequate to guarantee retrieval of the response image may by happenstance give the correct response image selection during association. This "luck" usually gives rise to response errors at a later time.

*This is the only use of a random variable in EPAM. We do not like it. We use it only because we have not yet discovered a plausible and satisfying adaptive mechanism for making the decision. The random mechanism does, however, give better results than the go-one-way-all-the-time mechanism which has also been used.

Second, suppose that the association building process does its job thoroughly. The cue which it builds is sufficient to retrieve the response image at one particular time, the time at which the two ite items were associated. If, at some future time, the net is grown to encompass new images being added to the memory, then a cue which previously was sufficient to correctly retrieve a response image may no longer be sufficient to retrieve that response image. In EPAM, association links are "dated," and ever vulnerable to interruption by further learning. Responses may be "unlearned" or "forgotten" temporarily, not because the response information has been destroyed in the memory, but because the information has been temporarily lost in a growing network. If an association failure of this type can be detected through feedback from the environmental or experimental situation, then the trouble is easily remedied by adding additional response information to the cue. If not, then the response may be more or less permanently lost in the net. The significance of this phenomenon will perhaps be more easily appreciated in the discussion of results of the EPAM simulation.

Responding: Internal and External

A conceptual distinction is made between the process by which EPAM selects an internal response image and the process by which it converts this image into an output to the environment.

Response retrieval. A stimulus item is presented. This stimulus input code is sorted in the discrimination net to retrieve the image list, in which the cue is found. The cue is sorted in the net to retrieve another image list containing the proposed response image. If there is no cue, or if on either sorting pass an empty image list is selected, no response is made.

Response generation. For purposes of response generation, there is a fixed discrimination net (decoding net), assumed already learned, which transforms letter codes of internal images into output form. The response image is decoded letter by letter by the net interpreter in the decoding net for letters.

The Organization of the Learning Task

The learning of nonsense symbols by the processes heretofore described takes time. EPAM is a serial machine. Therefore, the individual items must be dealt with in some sequence. This sequence is not arbitrarily prescribed. It is the result of higher order executive processes whose function is to control EPAM's focus of attention. These macroprocesses, as they are called, will not be described or discussed here. A full exposition of them is available in a paper by Feigenbaum and Simon.[13]

Stating the Model Precisely: Computer Program for EPAM

The EPAM model has been realized as a program in Information Processing Language V [14] and is currently being run both on the Berkeley 704 and the RAND 7090. Descriptive information on the computer realization, and also the complete IPL-V program and data structures for EPAM (as it stood in October, 1959) are given in an earlier work by the author [15].

IPL-V, a list processing language, was well suited as a language for the EPAM model for these key reasons:

a. The IPL-V basic processes deal explicitly and directly with list structures. The various information structures in EPAM (e.g., discrimination net, image list) are handled most easily as list structures. Indeed, the discrimination is, virtually by definition, a list structure of a simple type.

b. It is useful in some places, and necessary in others, to store with some symbols information descriptive of these symbols. IPL-V's description list and description list processes are a good answer to this need.

c. The facility with which hierarchies of subroutine control can be written in IPL-V makes easy and uncomplicated the programming of the kind of complex control sequence which EPAM uses.

Empirical Explorations with EPAM

The procedure for exploring the behavior of EPAM is straightforward. We have written an "Experimenter" program and we give to this program the particular conditions of that experiment as input at the beginning of an experiment. The Experimenter routine then puts EPAM qua subject through its paces in that particular experiment. The complete record of stimuli presented and responses made is printed out, as in the final net. Any other information about the processing or the state of the EPAM memory can also be printed out.

A number of simulations of the basic paired-associate and serial-anticipation experiments have been run. Simulations of other classical experiments in the rote learning of nonsense syllables have also been run. The complete results of these simulation experiments and a comparison between EPAM's behavior and the reported behavior of human subjects will be the subject of a later report. However, some brief examples here will give an indication of results expected and met.

a. Stimulus and response generalization. These are psychological terms used to describe the following phenomenon. If X and X' are similar stimuli, and Y is the correct response to the presentation of X; then if Y is given in response to the presentation of X', this is called stimulus generalization. Likewise, if Y and Y' are similar responses, and Y' is given in response to the presentation of X, this is called response generalization. Generalization is common to the behavior of all subjects, and is found in the behavior of EPAM. It is a consequence of the responding process and the structure of the discrimination net. For those "stimuli" are similar in the EPAM memory whose input codes are sorted to the same terminal; and one "response" is similar to another if the one is stored in the same local area of the net as the other (and hence response error may occur when response cue information is insufficient).

b. Oscillation and Retroactive Inhibition. We have described these phenomena in an earlier section.

Oscillation and retroactive inhibition appear in EPAM's behavior as consequences of simple mechanisms for discrimination, discrimination learning, and association. They were in no sense "designed into" the behavior. The appearance of rather complex phenomena such as these gives one a little more confidence in the credibility of the basic assumptions of the model.

These two phenomena are discussed together here because in EPAM they have the same origin. As items are learned over time, the discrimination net grows to encompass the new alternatives. Growing the net means adding new tests, which in turn means that more information will be examined in all objects being sorted. An important class of sorted objects is the set of cues. Cue information sufficient at one moment for a firm association may be insufficient at a later moment. As described above, this may lead to response failure. The failure is caused entirely by the ordinary process of learning new items. In the case of oscillation, the new items are items within a single list being learned. In the case of retroactive inhibition, the new items are items of the second list being learned in the same discrimination net. In both cases the reason for the response failure is the same. According to this explanation, the phenomena are first cousins (an hypothesis which has not been widely considered by psychologists).

In the EPAM model, the term <u>interference</u> is no longer merely <u>descriptive</u>--it has a precise and operational meaning. The process by which later learning interferes with earlier learning is completely specified.

c. Forgetting. The usual explanations of forgetting use in one way or another the simple and appealing idea that stored information is physically destroyed in the brain over time (e.g., the decay of a "memory trace," or the overwriting of old information by new information, as in a computer memory). Such explanations have never dealt adequately with the commonplace observation that all of us can remember, under certain conditions, detailed and seemingly unimportant information after very long time periods have elapsed. An alternative explanation, not so easily visualized, is that forgetting occurs not because of information destruction but because learned material gets lost and inaccessible in a large and growing association network.

EPAM forgets seemingly well-learned responses. This forgetting occurs as a direct consequence of later learning by the learning processes. Furthermore, forgetting is only temporary: lost associations can be reconstructed by storing more cue information. EPAM provides a mechanism for explaining the forgetting phenomenon in the absence of any information loss. As far as we know, it is the first concrete demonstration of this type of forgetting in a learning machine.

<u>Conclusion: A Look Ahead</u>

Verification of an information processing theory is obtained by simulating many different experiments and by comparing in detail specific qualitative and quantitative features of real behavior with the behavior of the simulation. To date, Mr. Simon and I have run a number of simulated experiments. As we explore verbal learning further, more of these will be necessary.

We have been experimenting with a variety of "sense modes" for EPAM, corresponding to "visual" input and "written" output, "auditory" input and "oral" output, "muscular" inputs and outputs. To each mode corresponds a perceptual input coding scheme, and a discrimination net. Associations-across-nets, as well as the familiar associations-within-nets, are now possible. Internal transformations between representations in different modes are possible. Thus, EPAM can "sound" in the "mind's ear" what it "sees" in the "mind's eye," just as all of us do so easily. We have been teaching EPAM to read-by-association, much as one teaches a small child beginning reading. We have only begun to explore this new addition.

The EPAM model has pointed up a failure shared by all existing theories of rote learning (including the present EPAM). It is the problem of whether association takes place between symbols or between tokens of these symbols. For example, EPAM cannot learn a serial list in which the same item occurs twice. It cannot distinguish between the first and second occurrence of the the item. To resolve the problem we have formulated (and are testing) processes for building, storing, and responding from chains of token associations.

References

1. Hull, C. L., C. I. Hovland, R. T. Ross, M. Hall, D. T. Perkins and F. B. Fitch, Mathematico-deductive Theory of Rote Learning, New Haven, Connecticut, Yale University Press, 1940.

2. Gibson, E. J., "A Systematic Application of the Concepts of Generalization and Differentiation to Verbal Learning," Psychol. Rev., Vol. 47, 1940, pp. 196-229.

3. Atkinson, R. C., "An Analysis of Rote Serial Learning in Terms of a Statistical Model," Indiana University, Doctoral Dissertation, 1954.

4. Stevens, S. S., ed., Handbook of Experimental Psychology, New York, Wiley, 1951.

5. Hull, C. L., "The Influence of Caffeine and Other Factors on Certain Phenomena of Rote Learning," J. Gen. Psychol., Vol. 13, 1935, pp. 249-273.

6. Feldman, J., "An Analysis of Predictive Behavior in a Two Choice Situation," Carnegie Institute of Technology, Doctoral Dissertation, 1959.

7. Newell, A., J. C. Shaw, and H. A. Simon, "Report on a General Problem Solving Program," Information Processing: Proceedings of the International Conference on Information Processing, UNESCO, Paris, June, 1959, pp. 256-264.

8. Lindsay, R., "The Reading Machine Problem," CIP Working Paper #33, Carnegie Institute of Technology (Graduate School of Industrial Administration), Pittsburgh, 1960.

9. Wickelgren, W., "A Simulation Program for Conservative Focusing," unpublished manuscript, University of California, Berkeley, January, 1961.

10. Bruner, J. S., J. J. Goodnow, and G. A. Austin, A Study of Thinking, New York, Wiley, 1956.

11. Selfridge, O. G., "Pattern Recognition and Modern Computers," Proceedings of the 1955 Western Joint Computer Conference, IRE, March, 1955.

12. Minsky, M., "Steps Toward Artificial Intelligence," Proceedings of the IRE, Vol. 49, No. 1, January, 1961, pp. 8-30.

13. Feigenbaum, E. and H. A. Simon, "A Theory of the Serial Position Effect," CIP Working Paper #14, Carnegie Institute of Technology (Graduate School of Industrial Administration), Pittsburgh, 1959.

14. Newell, A., F. M. Tonge, E. A. Feigenbaum, G. H. Mealy, N. Saber, B. F. Green, and A. K. Wolf, Information Processing Language V Manual (Sections I and II), Santa Monica, California, The RAND Corporation, P-1897 and P-1918.

15. Feigenbaum, E., An Information Processing Theory of Verbal Learning, Santa Monica, California, The RAND Corporation, P-1817, October, 1959.

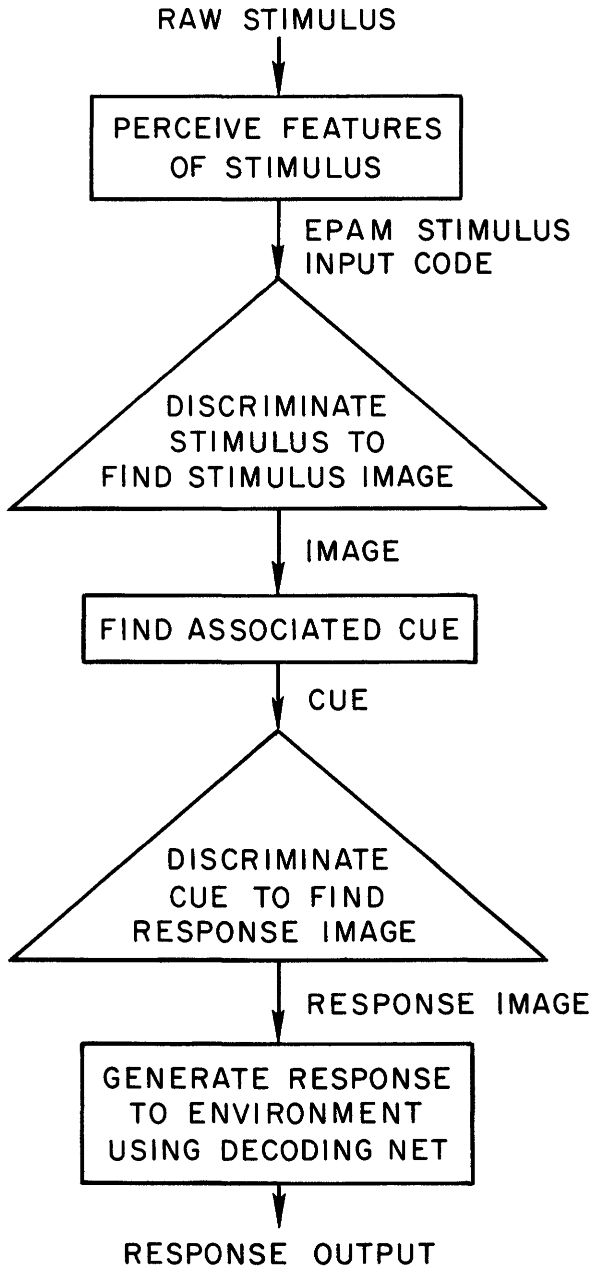

RAW STIMULUS
PERCEIVE FEATURES OF STIMULUS
EPAM STIMULUS INPUT CODE
DISCRIMINATE STIMULUS TO FIND STIMULUS IMAGE
IMAGE
FIND ASSOCIATED CUE
CUE
DISCRIMINATE CUE TO FIND RESPONSE IMAGE
RESPONSE IMAGE
GENERATE RESPONSE TO ENVIRONMENT USING DECODING NET
RESPONSE OUTPUT

Fig. 1—EPAM performance process for producing the response associated with a stimulus

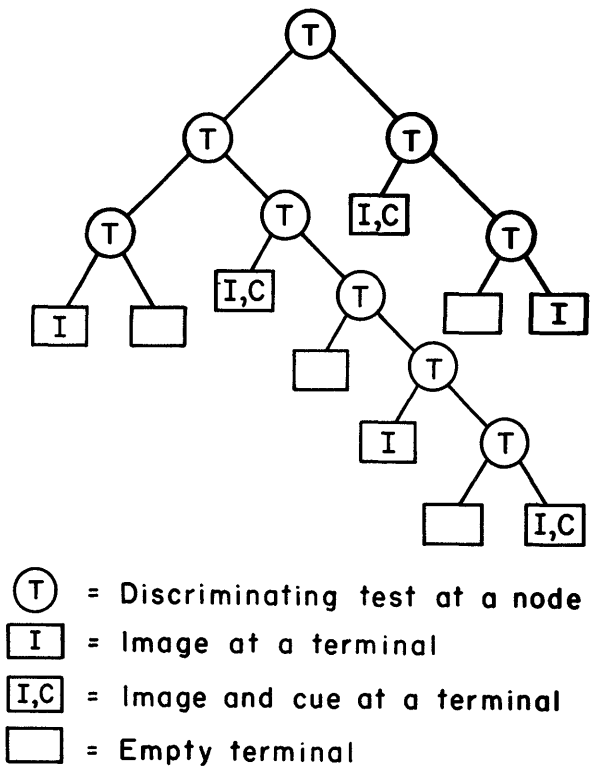

T
I,C
I
= Discriminating test at a node
= Image at a terminal
= Image and cue at a terminal
= Empty terminal

Fig. 2 — A typical EPAM discrimination net

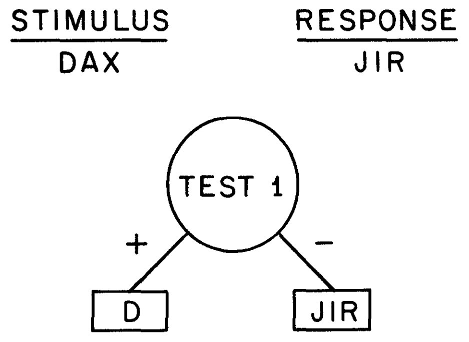


Fig. 3 — Discrimination net after the learning of the first two items. Cues are not shown. Condition: no redundant tests added. Test 1 is a first—letter test.

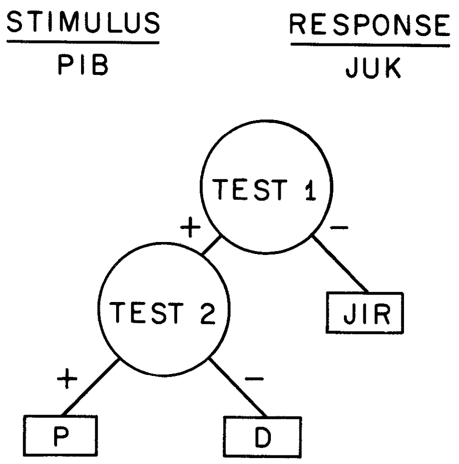


Fig. 4 — Discrimination net of Fig. 3 after the learning of stimulus item, PIB. Test 2 is a first letter test

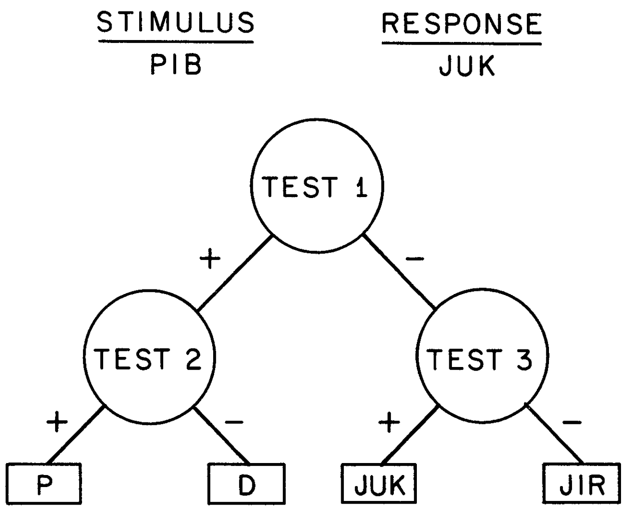


Fig. 5 — Discrimination net of Fig. 4 after the learning of the response item, JUK. Test 3 is a third—letter test

SIMULATION OF BEHAVIOR IN THE BINARY CHOICE EXPERIMENT

Julian Feldman
University of California, Berkeley

Summary

A modern, high-speed digital computer has been used to simulate the behavior of individual human subjects in a classical psychological experiment where the subject is asked to predict a series of binary events. The representation of models of human behavior in the form of computer programs has permitted the construction and study of more realistic hypothesis-testing models of behavior in this experiment rather than the over-simplified conditioning models previously proposed. A model for one subject is described in detail, and the problem of comparing the behavior of the model to the behavior of the subject is also discussed.

Introduction

Modern, high-speed digital computers have been used to simulate large, complex systems in order to facilitate the study of these systems. One of these systems that has been studied with the aid of computer simulation is man. The present paper describes another addition to the growing list of efforts to study human thinking processes by simulating these processes on a computer. The research summarized here has been concerned with simulating the behavior of individual subjects in the binary choice experiment.[3] The first section of this paper contains a description of the experiment. An overview of the model is given in the second section. The model for a particular subject is described in some detail in the third section.

The Binary Choice Experiment

In the binary choice experiment, the subject is asked to predict which of two events, E_1 or E_2, will occur on each of a series of trials. After the subject makes a prediction, he is told which event actually occurred. The sequence of events is usually determined by some random mechanism, e.g., a table of random numbers. One and only one event occurs on each trial. The events may be flashes of light or symbols on a deck of cards. The subject is usually asked to make as many correct predictions as he can.

In the research reported here, the experiment described in the preceding paragraph was modified by asking the subject to "think aloud"--to give his reasons for making a prediction as well as the prediction itself. The subject's remarks were recorded. The subject was instructed to "think aloud" in order to obtain more information on the processing that the subject was doing. This technique has been used in some of the classical investigations of problem-solving behavior[2,5] and in other computer simulation studies of thinking.[1,7] A comparison of the behavior of subjects in the binary choice experiment who did "think aloud" with the behavior of subjects who did not "think aloud" did not reveal any major differences.[3] The events in the present experiment were the symbols "plus" and "check." "Check" occurred on 142 of 200 trials and "plus" on the remaining 58 trials. The symbols were recorded on a numbered deck of 3 inch x 5 inch cards. After the subject made his prediction for trial t, he was shown card t which contained a "plus" or "check." While the subject was predicting the event of trial t, he could only see the event of trial t-1. A transcription of the tape recording of the remarks of subject DH and the experimenter, the author, in an hour-long binary choice experiment is presented in the Appendix. In the Appendix and the rest of this paper, the symbols "plus" and "check" are represented by "P" and "C" respectively. The transcription will be referred to as a protocol.

The Basic Model

To simulate the behavior of an individual subject in the binary choice experiment, a model of the subject's behavior must be formulated as a computer program. If the program is then allowed to predict the same event series as the subject has predicted, the behavior of the program--the predictions and the reasons--can be compared to the behavior of the subject. If the program's behavior is a reasonable facsimile of the subject's behavior, the program is at least a sufficient explanation of the subject's behavior. The level of explanation is really determined by the subject's statements. No attempt is made to go beyond these to more basic processes, e.g., neurological or chemical, of human behavior. Thus, the model is an attempt to specify the relationship between the reasons or hypotheses that the subject offers for his predictions and the preceding hypotheses, predictions, and events. The subject is depicted as actively proposing hypotheses about the structure of the event series. These hypotheses are tested by using them to predict events. If the prediction of the event is correct, the hypothesis is usually retained. If the prediction of the event is wrong, a new hypothesis is generally proposed.

The Model for DH

The model for each subject is based on a detailed examination of the protocol and some conjectures about human behavior. Perhaps the best thing to do at this point is to describe in some detail a model for the subject, DH, whose protocol appears in the Appendix.

The Hypotheses

This model proposes two types of hypotheses about the event series. The first type of hypothesis is a pattern of events. The model has a repertoire of nine patterns:

progression of C's
progression of P's
single alternation
2 C's and 1 P
1 C and 2 P's
2 P's and 2 C's
3 P's and 3 C's
4 P's and 4 C's
4 P's and 3 C's

The model can propose that the event series is behaving according to one of these patterns and use the pattern-hypothesis to predict the event of a given trial, t. The predictions of the first two patterns--progression of C's and progression of P's--for trial t are independent of the events preceding trial t. The predictions of the other patterns (the alternation patterns) are dependent on these preceding events. Thus, if the subject proposes the pattern "single alternation" for trial t and the event of trial t-1 was a C, the prediction for trial t is a P. In order to facilitate the determination of the prediction of an alternation pattern for trial t, the patterns are coded as sorting nets. For example, the pattern "2 C's and 1 P" is represented in the following fashion:

```
Is event t-1 a C?
     No--Predict C for trial t.
     Yes-Is event t-2 a C?
          No--Predict C for trial t.
          Yes-Predict P for trial t.
```

The second type of hypothesis that the model can propose is an anti-pattern or guess-opposite hypothesis. For example, the model can propose that the event of trial t will be the opposite of that predicted by a given pattern. This type of hypothesis is the model's representation of the notion of "gambler's fallacy"--the reason people predict "tails" after a coin falls "heads" seven times in a row.

The most general form of hypothesis has two components: a pattern component and a guess-opposite component. The prediction of the hypothesis is obtained by finding the prediction of the pattern component. If the hypothesis has a guess-opposite component, then the prediction of the hypothesis is the opposite of the pattern prediction. If the hypothesis does not have a guess-opposite component, then the prediction of the hypothesis is the prediction of the pattern component. Thus, while the prediction of the pattern-hypothesis "progression of C's" is always a C, the prediction of the hypothesis "guess-opposite-progression-of-C's" is always a P.

The Basic Cycle

The basic cycle of the model is as follows: The model uses an hypothesis to predict the event of trial t. The event is then presented. The model in Phase One "explains" the event of trial t with an explanation-hypothesis. In Phase Two a prediction-hypothesis for trial t+1 is formed. The model uses this prediction-hypothesis to predict trial t+1. The event of trial t+1 is presented, and the cycle continues.

Phase One

The basic motivation for this phase of the model is that the model must "explain" each event. An acceptable explanation is an hypothesis that could have predicted the event. The processing of Phase One is represented in the flow chart of Fig. 1.

The processing to determine the explanation-hypothesis for trial t begins by testing whether the pattern component of the prediction-hypothesis for trial t could have predicted the event of trial t correctly. If the pattern component could have predicted correctly, the pattern component is the explanation-hypothesis. If the pattern component could not have predicted correctly, the pattern-change mechanism is evoked. Thus if the prediction-hypothesis for trial t contained only a pattern component and the hypothesis predicted correctly, the explanation-hypothesis for trial t is the prediction-hypothesis for trial t. If the prediction-hypothesis for trial t contained a guess-opposite component and the hypothesis predicted correctly, the pattern-change mechanism is evoked because the pattern component could not have predicted the event correctly by itself. If the prediction-hypothesis for trial t was a guess-opposite-hypothesis and it predicted incorrectly, the pattern component of the prediction-hypothesis becomes the explanation-hypothesis for trial. t. The motivation here is really quite simple although the explanation may sound involved. If, in this binary situation, the hypothesis that a pattern will change leads to an incorrect prediction, the pattern must have persisted; and the pattern is an acceptable explanation of the event. If the hypothesis that a pattern will change leads to a correct prediction, the pattern obviously did not persist; and the possibility of a new pattern is considered.

The pattern-change mechanism is evoked on trial t if the pattern component of the prediction-hypothesis for trial t is unable to predict the event of trial t. The pattern-change mechanism consists of two parts. The first part evokes a subset of the nine patterns listed above. The second part of the pattern-change mechanism selects a single pattern out of the evoked set. A pattern is evoked, i.e., considered as a possible explanation of the event of trial t, if the pattern can predict the events of trials t and t-1. The pattern of the prediction-hypothesis for trial t, i.e., the pattern that cannot predict event t, is included in the evoked set if it can predict events t-1, t-2, and t-3. Of the patterns that are evoked, the pattern that has been selected most often on prior trials is selected as the pattern component of the explanation-hypothesis. If the pattern component of the prediction-hypothesis for trial t is selected, then the explanation-hypothesis is an anti-pattern hypothesis which is the model's interpretation of the subject's hypothesis "you have thrown me off the pattern" (cf. trial 9 of the protocol in the Appendix). The model interprets event t as an attempt to "throw me off" when the following three conditions are met: (1) the pattern is unable to predict the event of trial t; (2) the pattern is able to predict at least the three consecutive events of trials t-1, t-2, and t-3; and (3) the pattern is also the most frequently selected of those patterns that are evoked.

Phase Two

While Phase One is concerned mainly with the processing of the pattern-component of the hypothesis, Phase Two is concerned with the processing of the guess-opposite component. Phase Two is represented in the flow chart of Fig. 2.

If the prediction-hypothesis for trial t contained a guess-opposite component, the guess-opposite component is processed in a fashion quite analogous to the processing of the pattern component in Phase One. If the anti-pattern prediction-hypothesis for trial t predicted the event of trial t correctly, the guess-opposite component is retained, and the prediction-hypothesis for trial t+1 is guess-opposite-the-pattern-of-the-explanation-hypothesis. If the anti-pattern prediction-hypothesis for trial t predicted the event of trial t incorrectly, the guess-opposite component is considered for retention in a fashion analogous to the "throw-me-off" consideration for patterns. If the prediction-hypotheses for trials t-1 and t-2 had guess-opposite components and these hypotheses predicted correctly, then the guess-opposite component is retained for the prediction-hypothesis of trial t+1. If these conditions are not fulfilled, the guess-opposite component is dropped; and the prediction-hypothesis for trial t+1 is the explanation-hypothesis for trial t.

If the prediction-hypothesis for trial t did not contain a guess-opposite component, the model considers whether or not the guess-opposite component should be introduced on trial t+1. The model makes this decision on the basis of its past experience. It determines the number of consecutive events including and preceding the event of trial t that can be predicted by the explanation-hypothesis for trial t. This number will be called N_1. Then the model searches its memory backwards from the last trial included in N_1 to find a trial for which the explanation-hypothesis was the same as the explanation-hypothesis for trial t. Then the model determines the number of contiguous events including, preceding, and following this prior occurrence of the explanation-hypothesis of trial t that can be predicted by this hypothesis. This number will be called N_2.

If N_2=N_1, the model decides that the explanation-hypothesis for trial t will not be the prediction-hypothesis for trial t+1. The prediction-hypothesis for trial t+1 becomes guess-opposite-the-explanation-hypothesis for trial t. If $N_2 > N_1$, the model decides that the explanation-hypothesis for trial t will be the prediction-hypothesis for trial t+1. If $N_1 > N_2$, the model decides that this prior occurrence of the explanation-hypothesis for trial t is really not pertinent and continues to search its memory for an occurrence of the explanation-hypothesis where $N_2 \geqslant N_1$. If no such occurrence can be found, the prediction-hypothesis for trial t+1 is the explanation-hypothesis for trial t.

Predicting with the Models

Models of individual behavior like the one described for DH can be used to predict the same series of binary events that the subject was asked to predict. The predictions and hypotheses of the model--the model's protocol--can then be compared to the subject's protocol. The model does not speak idiomatic English, and so the comparison is made between the machine's protocol and a suitably coded version of the subject's protocol.

The model's protocol can be generated by presenting the model with the events in the same way the subject was presented with the events in the binary choice experiment; or the computer can take the experimenter's role, too, if suitable precautions are taken to prevent the model from peeking. However, this straightforward method of simulating the subject's behavior raises difficulties. These difficulties are identical to those of getting a chess or checker program to play a book game.[6,8] Because the decision of the chess or checker program at move m depends on its decisions at the preceding moves, m-1, m-2, ..., such a program, when it is playing a book game, must be "set back on the track" if its move deviates from the book move. The program and the book must have the same history if the program is to have a fair chance to make the same decision as that made in the book game. This "setting-back-on-the-track" may involve resetting a large number of parameters as well as changing the move itself. Elsewhere, I have called this "setting-back-on-the-track" technique <u>conditional prediction</u>.[4] The prediction of the model is conditional on the preceding decisions of the model being the same as those of the subject it is trying to predict.

The application of the conditional prediction technique to binary choice models such as the one described above for subject DH involves (1) comparing the program's behavior and the subject's behavior at every possible point, (2) recording the differences between the behaviors, and (3) imposing the subject's decision on the model where necessary. A type of monitor system is imposed on the program to perform these functions. The model for DH with the conditional prediction system controls is represented in Figs. 3 and 4. An example will help clarify these figures. In

Fig. 3, after each decision by the model to keep the pattern of the prediction-hypothesis for trial t for the explanation-hypothesis for trial t (B), this decision is compared to the subject's decision (1). If the model's decision was different from that of the subject, control is transferred to the pattern-change mechanism (3 trials). If the model's decision was the same as that of the subject, control is transferred to another part of the monitor (117 trials). Figs. 3 and 4 only contain the results for 195 trials because the model began at trial 6.

Conclusions

Deficiencies of the Models

The model for DH and the similar models that have been constructed to simulate the behavior of two other subjects in the binary choice experiment[3] are deficient in several respects. First of all, the comparison of the behavior of the model to that behavior of the subject from which the model was developed is, of course, not a very good test of the model. This type of comparison only yields some indication of the adequacy of the model and its components. Comparison of the behavior of the model to sequences of behavior of the subject not used in constructing the model awaits correction of some of the deficiencies mentioned below.

The segment of the model which has the highest number of errors relative to the number of times it is used is the guess-opposite segment (see Fig. 4). The subject certainly exhibits this type of behavior, but the model does not very often predict "guess opposite" when the subject does.

The pattern-change segment has a better error record, but it raises another issue. This segment is actually a selection device. A pattern is selected from the list of patterns that the subject uses. A more elegant pattern-change mechanism would generate a pattern out of the preceding sequence of events and some basic concepts. One of these concepts might be that patterns with equal numbers of P's and C's are preferred to alternation patterns with unequal numbers of P's and C's, all other things being equal.

The models have no mechanisms for making perceptual errors--"seeing" one symbol when another has occurred. Examination of the protocol of DH (Appendix) indicates that he does sometimes think that a C is a P (e.g., trial 196).

The models do not have a sufficiently rich repertoire of hypotheses. Subjects entertain more types of hypotheses about the event series than the two types, pattern and anti-pattern used in the model for DH. Some subjects entertain more sophisticated hypotheses. For example, one subject was able to detect the fact that a series of events was randomized in blocks of ten trials, i.e., the series had 7 P's and 3 C's in each block of ten trials.

Some evidence also exists that when suitably motivated by money, some subjects in a binary choice experiment will predict the most frequent event on each trial. Models for these subjects require statements of the conditions under which subjects abandon testing other hypotheses or at least abandon testing hypotheses by using them to predict events. Hypotheses could still be considered and tested without using them to predict events.

Contributions of the Models

The consequences of computer simulation for the study of human behavior have been discussed at some length in several places, and I have made a limited statement of my views on this matter in another place.[4] It will suffice then to discuss some of the implications of the work reported here for our understanding of behavior. The computer models of binary choice behavior are relatively simple computer programs; however, they are relatively complex psychological models. A widely accepted view of binary choice behavior has been the idea of verbal conditioning embodied in the stochastic learning model. In its simplest form, this model says the subject's probability of predicting E_1 or E_2 in the binary choice experiment is an exponentially-weighted moving average over preceding events. The verbal conditioning model is hardly consistent with the hypothesis-testing behavior exhibited by DH and a dozen other subjects for whom I have protocols. Protocols of group behavior in the binary choice experiment made available to me by David G. Hays are also consistent with the general idea of hypothesis-testing. Other inadequacies of the verbal conditioning model and evidence for hypothesis-testing models have been discussed elsewhere.[3]

The computer has provided the exponents of hypothesis-testing models of behavior with the means for studying and testing these complex models. Oversimplified explanations of human behavior can no longer be justified on the grounds that the means for studying complex models do not exist. Hopefully, the use of computers to simulate human behavior can extend man's intellect by helping him study his own behavior.

Acknowledgment

The author is indebted to Allen Newell for advice and suggestions made during the course of the research summarized in this paper.

References

1. Clarkson, G.P.E., and Simon, H.A. Simulation of individual and group behavior. American Economic Review, 1960, 50, 920-932.

2. Duncker, K. On problem solving. Psychological Monographs, 1945, 58, No. 270.

3. Feldman, J. An analysis of predictive behavior in a two-choice situation. Unpublished doctoral dissertation, Carnegie Institute of

Technology, 1959.

4. __________. Computer simulation of cognitive processes. in H. Borko (ed.), Computer applications in the behavioral sciences, Englewood Cliffs, Prentice-Hall, forthcoming.

5. Heidbreder, E. An experimental study of thinking. Archives of Psychology, 1924, 11, No. 73.

6. Newell, A., Shaw, J.C., and Simon, H.A. Report on the play of Chess Player I-5 of a book game of Morphy vs. Duke Karl of Brunswick and Count Isouard. CIP Working Paper No. 21, Graduate School of Industrial Administration, Carnegie Institute of Technology, 1959.

7. Newell, A., and Simon, H.A. The simulation of human thought. RAND Corporation Paper P-1734, 1959.

8. Samuel, A.L. Some studies in machine learning, using the game of checkers. IBM Journal of Research and Development, 1959, 3, 210-230.

Appendix: Protocol of Subject DH*

(All right, now I'll read the instructions to you. I'm going to show you a series of symbols. They will either be a P symbol or a C symbol. Before each word I'll give the signal NOW. When you hear the signal NOW, tell me what symbol you expect will occur on the next trial and why you selected that symbol. That's the purpose of the tape recorder. Take your time. After you have given me your guess, I will show you the correct symbol. Your goal is to anticipate each word as accurately as you can. Please... Well, do you have any questions?) Primarily, I just guess whether it'll be a P or a C. (That's it.) But this explaining why I think so. It can be little more than--I think it'll be this, I guess, I have a feeling. How more involved can it be than that? (Well, whatever reasons you have. If those are the only reasons that occur to you as you go thru this those will be the only reasons. Maybe they won't. OK, we'll try a few and then if you have any questions...)

(Now what do you expect the first symbol will be?) P. (OK, the 1st symbol is a C.)

(OK, now what do you expect the 2d symbol will be?) It'll be a P. (Why?) It's pictured in my mind. (OK, the 2d symbol is a C.)

I'll say a C. (Why?) Primarily this time because I'm trying to outguess you. (OK, the 3d symbol is a C.)

*The statements in parentheses are those of the experimenter.

(What do you say for the 4th symbol?) I'll say C again. (Why?) This time I feel it'll be a C. (The 4th symbol is a C. When you give your answer, if you say, "I think the 5th one will be something," it'll be easier to check the tape against the answer sheet.)

(What do you think the 5th one will be?) The 5th one will be a P. (Why is that?) I feel it'll be a P, that's all. (The 5th one is a C.)

(What do you think the 6th one will be?) The 6th one will be a C because you've been giving me C's all along, and I don't think this progression will end. (The 6th one is a C.)

(What do you think the 7th one will be?) The 7th one will be a C because I don't think the progression will be broken. (OK, the 7th one was a C.)

The 8th one will be a C for the same reason. You won't break the progression. (OK, the 8th one is a P.)

(What do you think the 9th one will be?) The 9th one will be a C. (Why is that?) I think that you just gave me the P to throw me off and you'll continue the progression. (The 9th one is a C. Oh, one thing, can you see these cards?) Yes. (Can you see me writing?) No, I can't. (OK.) I'm not looking. (Well, you can look at these cards. I want you to see I'm not picking these out of my head. This set has been predetermined.)

All right. This one will be a P. The 10th one will be a P. (Why is that?) I feel that the progression will start to mix up now. (The 10th one is a C.)

(What do you think the 11th one will be?) The 11th one will be a C. You're continuing the progression. (The 11th one is a C.)

(What do you think the 12th one will be?) The 12th one will be a C because you're continuing the progression. (The 12th one is a P.)

The 13th one will be a C. The 12th one was a P. You were trying to throw me off. The progression will continue. (The 13th one is a P.)

The 14th one will be a P. You're beginning a new progression with P's. (The 14th one is a P.)

The 15th one will be a P. You're still continuing the progression. (The 15th one is a P.)

(What about the 16th one?) The 16th one will be a C. ...to throw me off now. (The 16th one is a C.)

The 17th one will be a C. You're going to see if I'll revert to the progression of P's. (The 17th one is a C.)

The 18th one will be a P. You're going to break this progression of C's. (The 18th one is a C.)

The 19th one will be a P. You're going to get off this progression of C's. (The 19th one is a P.)

The 20th one will be a P. You're going to try to throw me off trying to make me think that all--think you're going back to the other progression which I'm confused about now. I don't remember what the last one was--C, I believe. (The 20th one is a P.)

The 21st one will be a C. You won't continue with the progression of P's. (The 21st one is a P.)

The 22d one is a C. You're doing this so that I might think the P progression will continue. (The 22d one is a C.)

The 23d one will be a C. You're trying to make me think that the next one will be a P--going back to the old progression. (The 23d one is a C.)

The 24th one will be a C. You're going to continue the progression of C's. (The 24th one was a C.)

The 25th one is a C. You're still going to continue the progression of C's. (The 25th one is a C.)

The 26th one is still a C. You'll continue the progression. (The 26th one is a C.)

The 27th one is a P. You'll break the progression now. (The 27th one is a C.)

The 28th one will be a P. You're going to break the progression now. (The 28th one is a C.)

The 29th one is a C. You're continuing the progression. (The 29th one is a C.)

The 30th is a C. You'll still continue the progression. (The 30th is a C.)

The 31st is a C. You'll continue the progression. (The 31st is a C.)

The 32d is a C. You'll still continue the progression. (The 32d is a P.)

The 33d is a C. You gave me a P last time to throw me off. (The 33d is a C.)

The 34th is a C. You'll continue the progression. (The 34th is a C.)

The 35th is a P. You're going to throw me off the progression. (The 35th is a C.)

The 36th is a C. You'll continue the progression. (The 36th is a C.)

The 37th is a C. You'll continue the progression. (The 37th is a C.)

The 38th is a C. You'll continue the progression. (The 38th is a C.)

The 39th is a C. You'll continue the progression. (The 39th is a C.)

The 40th is a C. You'll continue the progression. (The 40th is a C.)

The 41st is a C. You'll continue the progression. (The 41st is a C.)

The 42d is a C. You'll continue the progression. (The 42d is a C.)

The 43d is a C. You'll still continue the progression. (The 43d is a C.)

The 44th is a C. You'll still continue the progression. (The 44th is a C.)

The 45th is a C. You'll still continue the progression. (The 45th is a C.)

The 46th is a C. You'll still continue the progression. (The 46th is a C.)

The 47th will be a P. You'll now break the progression. (The 47th is a C.)

The 48th will be a C. You'll go back to the old progression. (The 48th is a C.)

The 49th is a C. You'll continue the progression. (The 49th is a C.)

The 50th is a C. You'll continue the progression. (The 50th is a P.)

The 51st will be a C. You gave me the P to throw me off. (The 51st is a P.)

The 52d is a P. You've begun a progression of P's. (The 52d is a C.)

The 53d is a P. You gave me a C to throw me off. (The 53d is a C.)

The 54th is a C. You'll continue the progression of C's. (The 54th is a C.)

The 55th is a C. You'll still continue the progression. (The 55th is a C.)

The 56th is a C. You'll continue the progression. (The 56th is a P.)

57 is a P. The P will throw me off the progression thinking you had tried to throw me off the C progression with your last P. (57 you said was a P?) P. (57 was a C.)

58 is a C. You began a progression of C's. (58 is a P.)

59 is a C. You're still trying to throw me off with the C's. (59 is a P.)

60 will be a P. You're beginning a progression of P's. (60 is a C.)

61 is a P. You're zigzagging between P's and C's. (61 is a P.)

62 is a C. You'll continue the oscillation. (62 is a C.)

63 is a C--rather 63 is a P because of the oscillation pattern. (63 is a P.)

64 is a C becuase of the oscillation pattern. (64 is a C.)

65 is a P because of the oscillation pattern. (65 is a C.)

66 is a C. You've begun a progression of C's. (66 is a P.)

67 will be a C. You're oscillating again. (67 is a C.)

68 is a C. You're having a different type of oscillation--2 C's between a P. (68 is a P.)

69 is a C. You're oscillating with C's and P's. (69 is a C.)

70 will be a P. It's the alternate symbol. (70 is a P.)

71 will be a C because of the oscillation sequence. (71 is a C.)

72 will be a P because of the oscillation sequence. (72 is a C.)

73 will be a C. You've begun a new progression of C's. (73 is a C.)

74 is a C. You're continuing the progression. (74 is a C.)

75 is a C. You're still continuing with the progression. (75 is a C.)

76 is still a C. You're continuing with the progression. (76 is a C.)

77 is a C. You're still continuing with the progression. (77 is a C.)

78 is a C. The progression is continuing. (78 is a P.)

79 is a C. The P is to throw me off. The progression continues. (79 is a C.)

80 is a C. The progression will continue. (80 is a C.)

81 is a C. The progression continues. (81 is a P.)

82 will be a C. You're alternating now with C's and P's. (82 is a P.)

83 is a P. You've begun a progression of P's. (83 is a C.)

84 will be a C. The P's were given to throw me off. (84 is a P.)

85 will be a P. You've begun a new alternating sequence. (85 is a P.)

86 will be a C. You're following with a C and 2 P's. Another C will come. (86 is a C.)

87 will be a P. You'll follow the same sequence. (87 is a C.)

88 will be a P. You've begun a sequence of 2 C's and a P. (88 is a C.)

89 is a C. You've begun a new progression of C's. (89 is a C.)

90 is a C. You'll continue the progression. (90 is a C.)

91 is a C. The progression continues. (91 is a C.)

92 is a C. The progression continues. (92 is a P.)

93 is a P. The P's given to me previously to make me think that the progression was being broken and that you would revert to it after the P. The next one will be a P. (93 is a C.)

94 will be a C. You've gone back to the C progression. (94 you say now is a C.) 94 is a C. (OK, 94 is a C.)

95 is a C. You've begun a progression of C's. (95 is a P.)

96 will be a C. You're alternating now with C's and P's. (96 is a P.)

97 is a C. You've begun a progression of a C and 2 P's. (97 is a P.)

98 is a P. You've begun a progression of P's. (98 is a C.)

99 is a C. You've begun a progression of 3 P's and 3 C's. You've already had the 3 P's. 98 (sic) will be a C. (That was...99 is going to be a C. You said. 99 is a C.)

(What's 100?) 100 will be a C. It follows the progression. (100 is a C.)

101 will still be a C. Continue the progression of 3 P's and 3 C's. (101 is a C.)

102 will be a C. You've begun a progression of C's. (102 is a C.)

103 is a C. You'll continue the progression of C's. (103 is a C.)

104 is a C. You'll continue with the progression. (104 is a C.)

105 will be a C. You'll continue the progression. (105 is a C.)

106 will be a P. You'll break the progression now. (106 was a C.)

107 will be a C. You'll continue the progression. (107 was a P.)

108 will be a C. You gave me the P to throw me off. The progression will continue. (108 is a C.)

109 will be a C. You'll continue the progression. (109 was a P.)

110 will be a C. You're alternating with C's and P's. (110 is a C.)

111 will be a P. You'll continue the alternation. (111 was a P.)

112 will be a C. You'll continue the alternation. (112 was a P.)

113 will be a C. You've begun a progression of a C and 2 P's. (113 is a P.)

114 will be a P. You've begun a progression of P's. (114 is a P.)

115 will be a P. You'll continue the progression. (115 is a C.)

116 will be a P. The C was given to throw me off. (116 is a C.)

117 is a C. You've begun a progression of 4 P's and 4 C's. (117 is a P.)

118 will be a P. The progression has changed from 4 P's and 4 C's to 4 P's and 3 C's. (118 is a C.)

119 will be a P. You're alternating with C's and P's. (119 is a C.)

120 will be a C. You're continuing the progression. (120 is a P.)

121 will be a P. You have a progression of 2 C's and 2 P's. (121 is a P.)

122 will be a C. You'll continue this progression of 2 and 2. (122 is a C.)

123 will be a C. You're continuing the progression. (Of what?) Of 2 C's and 2 P's. (123 is a C.)

124 will be a C. You've begun a progression of C's. (124 is a C.)

124 (sic) will be a C. You're continuing the progression. (125 is a C.)

126 will be a C. You're continuing the progression. (126 is a P.)

127 will be a C. You gave me the P to throw me off. (127 is a P.)

128 will be a P. You've begun a progression of P's. (128 is a C.)

129 will be a C. You've begun a progression of 2 P's and 2 C's. (129 was a C.)

130 will be a C. You've begun a progression of C's. (130 is a P.)

131 will be a P. You're continuing the progression of 2 P's and 2 C's. (131 is a C.)

132 will be a P. You're alternating the signs now. (132 is a C.)

133 will be a C. You've begun a sequence of C's. (133 is a C.)

134 will be a C. You're continuing the sequence. (134 is a C.)

135 is a C. You're continuing with the progression. (135 is a P.)

136 will be a P. You've begun...you're trying to throw me off now with a 2d P. Think there would be only one P. (136 is a C.)

137 is a C. You're going to continue with the progression of C's. (137 is a C.)

138 is a C. You'll continue the progression. (138 is a C.)

139 is a C. You'll continue the progression. (139 is a P.)

140 is a C. The P was given to throw me off. (140 is a P.)

141 is a C. You gave me the 2 C's (sic) for the same reason as the previous time you had given me the 2 C's 'er 2 P's... (141 is a C.)

142 is a C. You'll continue with the progression. (142 is a C.)

143 is a C. You'll continue with the progression. (143 is a C.)

144 is a C. You'll continue with the progression. (144 is a C.)

145 is a P. You'll break the progression. (145 is a C.)

146 is a C. You'll continue the progression. (146 is a C.)

147 is a C. You'll continue the progression. (147 is a C.)

148 is a C. You'll continue the progression. (148 is a C.)

149 is a C. You'll continue the progression. (149 is a C.)

150 is a C. You'll still continue the progression. (150 is a C.)

151 is a C. You'll still continue the progression. (151 is a C.)

152 will be a P. You'll break the progression. (152 is a C.)

153 is a C. You'll continue the progression. (153 is a P.)

154 is a C. You've broken the progression and you'll revert to it now. (154 is a C.)

155 is a C. You'll continue the progression. (155 is a P.)

156 is a C. You're alternating with P's and C's. (156 is a C.)

157 is a C. The alternation of P's and C's was to throw me off the progression of C's. The C progression will continue. (157 is a P.)

158 is a C. You're still going back to C sequence. (158 is a C.)

159 is a C. You're still going to continue this sequence. (159 is a P.)

160 is a C. You have an alternating sequence of P's and C's. (160 is a C.)

161 will be a P. You'll continue to alternate. (161 is a P.)

162 will be a C. You'll continue this oscillation. (162 is a P.)

163 will be a C. You'll continue the alternation. (163 is a C.)

164 will be a P. You'll continue the alternation. (164 is a C.)

165 will be a P. You'll go back to the alternation. (165 is a C.)

166 will be a C. You've begun a sequence of C's. (166 is a C.)

167 will be a C. You've begun a sequence of C's. (167 is a P.)

168 will be a P. You've begun a sequence of 2 C's and 2 P's. (168 is a C.)

169 is a C. The previous P's were given to throw me off. You'll continue the sequence of C's. (169 is a C.)

170 will be a C. You'll continue the sequence. (170 is a P.)

171 will be a P. You'll begin a sequence of P's. (171 is a P.)

172 will be a C. You'll revert to the C's. (172 is a C.)

173 will be a C. You're alternating with 2 P's and 2 C's. (173 is a P.)

174 will be a C. The alternation is a C and a P. (174 is a C.)

175 will be a P. You'll continue this alternation. (175 is a C.)

176 will be a C. You've begun a sequence of C's. (176 is a P.)

177 will be a C. You'll continue with the progression of C's. (177 is a P.)

178 will be a C. You've begun a progression of 2 P's and 2 C's. (What did you say 178 was?) A C. (178 is a C.)

179 will be a C. You'll continue with another C to complete the sequence of 2 P's and 2 C's. (179 is a C.)

180 will be a P. You'll continue this sequence. (180 is a C.)

181 is a C. You've begun a sequence of C's. (181 is a C.)

182 is a C. You'll continue the sequence. (182 is a C.)

183 is a C. You'll continue the sequence. (183 is a P.)

184 will be a C. The P was given to throw me off. (184 is a C.)

185 is a C. You'll continue the sequence of C's. (185 is a C.)

186 will be a C. You'll continue the sequence. (186 is a C.)

187 will be a C. You'll continue the sequence. (187 is a C.)

188 is a C. You'll continue the sequence. (188 is a C.)

189 is a C. You'll continue the sequence. (189 is a P.)

190 will be a C. The P was given to throw me off. (190 is a C.)

191 will be a C. The double P (sic) was given to throw me off a little more. (191 is a C.)

192 is a C. You've...been giving me a sequence of 2 P's and 2 C's. (192 is a C.)

192 (sic) is a P. You're continuing the sequence of 2 P's and 2 C's. (193 is a C.)

194 is a C. You've begun a sequence of C's. (194 is a C.)

195 is a C. You'll continue the sequence. (195 is a P.)

196 will be a P. You have a sequence here of inserting 2 P's. (196 is a C.)

197 is a C. The P was given to throw me off. (197 is a C.)

198 will be a C. You'll continue the sequence. (198 is a C.)

199 is a C. You'll continue the sequence. (199 is a C.)

200 will be a C. You'll continue the sequence. (200 is a C.)

```
A. COULD THE PATTERN COMPONENT OF THE PREDICTION-HYPOTHESIS
   FOR TRIAL T HAVE PREDICTED THE EVENT OF TRIAL T CORRECTLY.
   B. YES-EXPLANATION-HYPOTHESIS FOR TRIAL T IS THE PATTERN
      COMPONENT OF THE PREDICTION-HYPOTHESIS FOR TRIAL T.
   C. NO--EVOKE PATTERNS THAT COULD HAVE PREDICTED THE EVENTS
      OF TRIALS T AND T-1 CORRECTLY. THE PATTERN OF THE
      PREDICTION-HYPOTHESIS FOR TRIAL T IS EVOKED IF IT COULD
      HAVE PREDICTED CORRECTLY THE EVENTS OF TRIALS T-1,T-2,
      AND T-3.
   D. SELECT FROM THE SET OF EVOKED PATTERNS THAT PATTERN THAT
      HAS BEEN SELECTED MOST OFTEN ON PRECEDING TRIALS.
   E. IS THE SELECTED PATTERN THE PATTERN COMPONENT OF THE
      PREDICTION-HYPOTHESIS FOR TRIAL T.
      F. YES-EXPLANATION-HYPOTHESIS FOR TRIAL T IS THROW ME OFF
         THE SELECTED PATTERN.
      G. NO--EXPLANATION-HYPOTHESIS FOR TRIAL T IS THE
         SELECTED PATTERN.
```

FIG. 1. PHASE ONE OF BINARY CHOICE MODEL FOR DH.

```
H. DID THE PREDICTION-HYPOTHESIS FOR TRIAL T CONTAIN A GUESS-
   OPPOSITE COMPONENT.
   I. YES-DID THE PREDICTION-HYPOTHESIS FOR TRIAL T PREDICT
      THE EVENT OF TRIAL T CORRECTLY.
      J. YES-PREDICTION-HYPOTHESIS FOR TRIAL T+1 IS GUESS-
         OPPOSITE THE PATTERN COMPONENT OF THE EXPLANATION-
         HYPOTHESIS FOR TRIAL T.
      K. NO--DID THE PREDICTION HYPOTHESES FOR TRIALS T-1 AND
         T-2 CONTAIN GUESS-OPPOSITE COMPONENTS AND WERE THE
         PREDICTIONS OF THE EVENTS OF THESE TRIALS CORRECT.
         L. YES-PREDICTION-HYPOTHESIS FOR TRIAL T+1 IS GUESS-
            OPPOSITE THE EXPLANATION-HYPOTHESIS FOR TRIAL T.
         M. NO--PREDICTION-HYPOTHESIS FOR TRIAL T+1 IS THE
            EXPLANATION-HYPOTHESIS FOR TRIAL T.
   N. WILL THE EXPLANATION-HYPOTHESIS FOR TRIAL T CONTINUE.
      (SEE TEXT FOR AN EXPLANATION OF THIS TEST)
      O. YES-PREDICTION-HYPOTHESIS FOR TRIAL T+1 IS THE
         EXPLANATION-HYPOTHESIS FOR TRIAL T.
      P. NO--PREDICTION-HYPOTHESIS FOR TRIAL T+1 IS GUESS-
         OPPOSITE THE EXPLANATION-HYPOTHESIS FOR TRIAL T.
Q. PREDICT EVENT FOR TRIAL T+1.
```

FIG. 2. PHASE TWO OF BINARY CHOICE MODEL FOR DH.

```
A. COULD THE PATTERN COMPONENT OF THE PREDICTION-HYPOTHESIS
   FOR TRIAL T HAVE PREDICTED THE EVENT OF TRIAL T CORRECTLY.
   B. YES-EXPLANATION-HYPOTHESIS FOR TRIAL T IS THE PATTERN
      COMPONENT OF THE PREDICTION-HYPOTHESIS FOR TRIAL T.
                   * 1. DID SUBJECTS EXPLANATION-HYPOTHESIS
                   *    FOR TRIAL T CONTAIN PATTERN COMPONENT
                   *    OF THE PREDICTION-HYPOTHESIS FOR TRIAL T.  120
                   *    YES-GO TO 6.                               117
                   *    NO--ERROR--FAILURE TO EVOKE PATTERN-
                   *    CHANGE MECHANISM. GO TO C.                   3
   C. NO--EVOKE PATTERNS THAT COULD HAVE PREDICTED THE EVENTS
      OF TRIALS T AND T-1 CORRECTLY. THE PATTERN OF THE
      PREDICTION-HYPOTHESIS FOR TRIAL T IS EVOKED IF IT COULD
      HAVE PREDICTED CORRECTLY THE EVENTS OF TRIALS T-1,T-2,
      AND T-3.
                   * 2. WAS THE PATTERN OF THE SUBJECTS EXPLA-
                   *    NATION-HYPOTHESIS FOR TRIAL T EVOKED.       78
                   *    YES-GO TO D.                                61
                   *    NO--ERROR--FAILURE TO EVOKE PATTERN.
                   *    ADD SUBJECTS PATTERN TO EVOKED SET
                   *    AND CONTINUE.                               17
   D. SELECT FROM THE SET OF EVOKED PATTERNS THAT PATTERN THAT
      HAS BEEN SELECTED MOST OFTEN ON PRECEDING TRIALS.
                   * 3. WAS THE PATTERN OF THE SUBJECTS EXPLA-
                   *    NATION-HYPOTHESIS FOR TRIAL T SELECTED.     78
                   *    YES-GO TO E.                                64
                   *    NO--ERROR--FAILURE TO SELECT PATTERN.
                   *    REPLACE INCORRECT PATTERN WITH SUBJECTS
                   *    PATTERN AND CONTINUE.                       14
   E. IS THE SELECTED PATTERN THE PATTERN COMPONENT OF THE
      PREDICTION-HYPOTHESIS FOR TRIAL T.
      F. YES-EXPLANATION-HYPOTHESIS FOR TRIAL T IS THROW ME OFF
         THE SELECTED PATTERN.
                   * 4. DID SUBJECTS EXPLANATION-HYPOTHESIS
                   *    FOR TRIAL T CONTAIN THROW-ME-OFF.           27
                   *    YES-GO TO H.                                26
                   *    NO--ERROR--INCORRECT EVOCATION OF
                   *    THROW-ME-OFF. DELETE THROW-ME-OFF
                   *    AND GO TO H.                                 1
      G. NO--EXPLANATION-HYPOTHESIS FOR TRIAL T IS THE
         SELECTED PATTERN.
                   * 5. DID SUBJECTS EXPLANATION-HYPOTHESIS
                   *    FOR TRIAL T CONTAIN THROW-ME-OFF.           51
                   *    YES-ERROR--FAILURE TO EVOKE THROW-ME-
                   *    OFF. INSERT THROW-ME-OFF AND GO TO H.        3
                   *    NO--GO TO H.                                48
                   * 6. DID SUBJECTS EXPLANATION-HYPOTHESIS
                   *    FOR TRIAL T CONTAIN THROW-ME-OFF.          117
                   *    YES-ERROR--FAILURE TO EVOKE THROW-ME-
                   *    OFF. INSERT THROW-ME-OFF AND GO TO H.        3
                   *    NO--GO TO H.
```

FIG. 3. SUMMARY OF BEHAVIOR OF PHASE ONE OF BINARY CHOICE MODEL FOR DH ADAPTED FOR CONDITIONAL PREDICTION.

```
  H. DID THE PREDICTION-HYPOTHESIS FOR TRIAL T CONTAIN A GUESS-
     OPPOSITE COMPONENT.
     I. YES-DID THE PREDICTION-HYPOTHESIS FOR TRIAL T PREDICT
        THE EVENT OF TRIAL T CORRECTLY.
```

FIG. 4. SUMMARY OF BEHAVIOR OF PHASE TWO OF BINARY CHOICE MODEL FOR DH ADAPTED FOR CONDITIONAL PREDICTION.

```
              J. YES-PREDICTION-HYPOTHESIS FOR TRIAL T+1 IS GUESS-
                 OPPOSITE THE PATTERN COMPONENT OF THE EXPLANATION-
                 HYPOTHESIS FOR TRIAL T.
                             * 7. DID SUBJECTS PREDICTION-HYPOTHESIS
                             *    FOR TRIAL T+1 CONTAIN GUESS-OPPOSITE
                             *    COMPONENT.                                     6
                             *    YES-GO TO 12.                                  5
                             *    NO--ERROR--INCORRECT RETENTION OF
                             *    GUESS-OPPOSITE COMPONENT. DELETE
                             *    GUESS-OPPOSITE AND GO TO 12.                   1
              K. NO--DID THE PREDICTION HYPOTHESES FOR TRIALS T-1 AND
                 T-2 CONTAIN GUESS-OPPOSITE COMPONENTS AND WERE THE
                 PREDICTIONS OF THE EVENTS OF THESE TRIALS CORRECT.
                 L. YES-PREDICTION-HYPOTHESIS FOR TRIAL T+1 IS GUESS-
                    OPPOSITE THE EXPLANATION-HYPOTHESIS FOR TRIAL T.
                             * 8. DID SUBJECTS PREDICTION-HYPOTHESIS
                             *    FOR TRIAL T+1 CONTAIN GUESS-OPPOSITE
                             *    COMPONENT.                                     2
                             *    YES-GO TO 12.                                  2
                             *    NO--ERROR--INCORRECT RETENTION OF
                             *    GUESS-OPPOSITE COMPONENT. DELETE
                             *    GUESS-OPPOSITE AND GO TO 12.                   0
                 M. NO--PREDICTION-HYPOTHESIS FOR TRIAL T+1 IS THE
                    EXPLANATION-HYPOTHESIS FOR TRIAL T.
                             * 9. DID SUBJECTS PREDICTION-HYPOTHESIS
                             *    FOR TRIAL T+1 CONTAIN GUESS-OPPOSITE
                             *    COMPONENT.                                    12
                             *    YES-ERROR--FAILURE TO KEEP GUESS-
                             *    OPPOSITE COMPONENT. INSERT GUESS-
                             *    OPPOSITE AND GO TO 12.                         1
                             *    NO--GO TO 12.                                 11
           N. WILL THE EXPLANATION-HYPOTHESIS FOR TRIAL T CONTINUE.
              (SEE TEXT FOR AN EXPLANATION OF THIS TEST)
              O. YES-PREDICTION-HYPOTHESIS FOR TRIAL T+1 IS THE
                 EXPLANATION-HYPOTHESIS FOR TRIAL T.
                             *10. DID SUBJECTS PREDICTION-HYPOTHESIS
                             *    FOR TRIAL T+1 CONTAIN GUESS-OPPOSITE
                             *    COMPONENT.                                   136
                             *    YES-ERROR--FAILURE TO EVOKE GUESS-
                             *    OPPOSITE COMPONENT. INSERT GUESS-
                             *    OPPOSITE AND GO TO 12.                        10
                             *    NO--GO TO 12.                                126
              P. NO--PREDICTION-HYPOTHESIS FOR TRIAL T+1 IS GUESS-
                 OPPOSITE THE EXPLANATION-HYPOTHESIS FOR TRIAL T.
                             *11. DID SUBJECTS PREDICTION-HYPOTHESIS
                             *    FOR TRIAL T+1 CONTAIN GUESS-OPPOSITE
                             *    COMPONENT.                                    39
                             *    YES-GO TO 12.                                  2
                             *    NO--ERROR--INCORRECT SELECTION OF
                             *    GUESS-OPPOSITE. DELETE GUESS-OPPOSITE
                             *    AND CONTINUE.                                 37

                             *12. WAS PATTERN OF SUBJECTS EXPLANATION-
                             *    HYPOTHESIS FOR TRIAL T THE SAME AS
                             *    THE PATTERN OF THE SUBJECTS PREDICTION-
                             *    HYPOTHESIS FOR TRIAL T+1.                    195
                             *    YES-GO TO Q.                                 192
                             *    NO--ERROR--FAILURE TO CHANGE PATTERN.
                             *    INSERT SUBJECTS PATTERN IN PREDICTION-
                             *    HYPOTHESIS FOR TRIAL T+1 AND CONTINUE.         3
        Q. PREDICT EVENT FOR TRIAL T+1.
                             *13. DID SUBJECT PREDICT SAME EVENT.              195
                             *    YES-GO TO A.                                 193
                             *    NO--ERROR--INCORRECT PREDICTION.
                             *    CORRECT AND GO TO A.                           2
```

FIG. 4. SUMMARY OF BEHAVIOR OF PHASE TWO OF BINARY CHOICE MODEL FOR DH ADAPTED FOR CONDITIONAL PREDICTION (CONTINUED).

PROGRAMMING A MODEL OF HUMAN CONCEPT FORMULATION

Earl B. Hunt and Carl I. Hovland
Yale University

Summary

A model of human information processing during concept formation has been constructed, using a list processing, digital computer program. The program's input consists of descriptions of objects in terms of dimensions and values. The universe of objects is divided into two or more sets. The program attempts to form a decision rule, based upon the descriptions of the objects, which can be used to assign any previously presented or new object to its correct set.

The program is a model for human information processing, rather than an artificial intelligence system. It contains features which limit the number of objects in internal memory and the number of dimensions which may be involved in an answer. Using this program, simulations have been performed of a number of psychological experiments in concept learning. Comparison of these simulations with the data obtained from human subjects will be discussed.

What is a concept? Ordinary usage is not precise. The English "the concept of force", "the concept of massive retaliation," and the "concept of dogs" are all permissible. Church[3] has offered a definition which has been accepted, implicitly, by psychologists who perform "concept learning" experiments. Church's argument is that any given symbol (or name) can be attached to the members of a set of objects. For any arbitrary object there exists a rule concerning the description of the object, a rule which can be used to decide whether or not the object is a member of the set of objects to which the name applies. The decision rule is the concept of the name, the set of objects is the denotation of the name.

In a typical concept learning experiment the subject is confronted with a series of stimuli which are given a particular name and another series of stimuli which are either given another particular name, or a series of different names. Thus the first set might be called "dogs" and the second either "not-dogs" or "cats, wolves, sheep, etc." Thus some routines are necessary to classify the instances to correspond to the names assigned by the experimenter. These are our ordering routines. Sometimes the various stimuli given one name have certain common characteristics, e.g. all the positive instances may have three triangles. At other times there are no common stimulus elements, but there are common relationships, e.g. all the positive instances may have the same size of upper figure as lower figure, although the figures may be large, medium or small sized in each row. A machine routine may be required to describe relations between basic stimulus elements. So we must have description routines in a simulation. Finally, different types of stimulus sets may be organized differently in terms of different types of logical connectives. Sometimes the concept involves the joint presence of two or more characteristics. Such concepts are referred to as conjunctive concepts (e.g. large red figure). Other concepts involve the presence of different subsets of characteristics. These are disjunctive concepts, e.g. red or large figures. Different ways of defining the form of an answer are provided by a set of solution routines.

The program must be capable of simulating a variety of conditions under which experiments have been performed. As illustrations of some of the variations, or manipulations, which must be simulated the following may be mentioned.

The number and complexity of the stimuli may vary from study to study. The speed of presentation of new stimuli can be altered. The instances may be left in view or the subject may be forced to rely on his memory. Different concept learning problems can be compared along such dimensions as: logical complexity of the correct answer, number of relevant vs. number of irrelevant dimensions, order of presentation of problems of different types, and presentation of information by positive or negative instances.

The subject may make a variety of responses during the experiment. Subjects may describe, verbally, their intermediate and final hypotheses concerning the characteristics of the concept. These responses may give us clues as to the nature of the individual's information processing procedures. As such, they constitute accessory measures used in our simulation studies. The time taken to develop an answer under various experimental conditions is also a useful response measure. The errors that subjects make in subsequent identifications of stimulus names can be analyzed. The more objective records are to be preferred, and our major goal is to predict these by computer simulation.

In order to develop a theoretical explanation of concept learning we have accepted the "black box" analogy. We have attempted to write a computer program which, when given as input coded representations of the stimuli, will give as output coded responses that can be used to predict the responses of a human subject. Accurate prediction of the responses, not the development of a good hypothesis developer, nor, solely, the reproduction of previously obtained protocols, is our goal. We are not concerned with the processes specific to the task of categorizing geometric patterns. These are used as stimuli because they are convenient and because they represent stimuli which can be described in terms of previously discriminated stimuli. Hopefully we shall be able to make conclusions about concept learning processes irrespective of the particular form of the stimuli.

Reports of our psychological experimental work have been, and are being, made in separate publications. This paper will be concerned with the programming details of our concept learning model. This model has been completed, debugged, and used to simulate several experiments. After describing the model we shall indicate the result of some simulations and discuss the modifications of the model which have been indicated.

The concept learning program is a list processing language program written for the IBM 709-7090 data processing systems. The original programs were written in Information Processing Language V (IPL-V), the interpreter list processing language developed by Newell, Simon, and Shaw (cf. Green[5]). Partly for local administrative reasons, we are in the process of converting our programs to LISP, a list processing language developed by McCarthy[14]. We do not have sufficient experience with LISP to compare the two languages for our type of problem. As the basic logic of the LISP and IPL-V programs are the same no distinction will be made between them.

Description of the Program

The program consists of two blocks of data, specified by the programmer at the beginning of each run, and five subsystems for data processing. At the beginning of a simulation the programmer specifies a sequence of problems, a set of parameters, and a set of lists. The last two represent the capabilities of the artificial subject. The problem data remains constant throughout the run, the specifications of the subject may be changed by the program.

Problems are presented by describing instances, the denotations of names, (classes), and the conditions of presentation to be used. This takes the form of specification of memory requirements, number of stimuli presented at a single time, etc. All the conditions used to describe a problem are specified in the property list of the symbol naming the problem.

Each instance (i.e., object to be categorized) is represented by a symbol whose property list specifies the symbol's class membership, and, by a list of pairs, the dimensions and values which constitute a formal description of the object. For instance, in our previous example, a large, red triangle would contain the following pairs on its description list: (class name-"alpha"), (size-large), (color-red), (shape-triangle). The formal description list constitutes the most molecular information about objects which is made available to the program. Higher order, working descriptions based upon relations between elements of the formal description may be developed by the program.

Dimensions represent the manner in which objects are free to vary. We have utilized a "dimensional analysis" of objects which specifies a finite universe with a built in structure to describe objects (cf. Hovland[8]). Dimensions are also organized into "dimension sets", or groupings. These groupings represent subsets of the sets of all dimensions which will be considered together during recognition and answer development.

The "subject" specifications fall into two broad categories; numerical parameters and initial settings used to control the program. They will be discussed as they enter into the action of the model.

Figure 1 specifies the channels of communications between the various subsystems in the model. There are two major groups of subsystems. The first, as indicated in Figure 1, is the recognition and memory system. Its task is to acquire information from the formal description of presented instances and to retain this information for later processing by the answer development and checking group.

By examining the property list of the problem, the program determines the conditions of presentation of stimuli. If these conditions would not require memorization by a human subject (i.e. if instances are presented to the subject and left in view) the name of each instance, together with its entire formal description, is added to internal memory as the instance is presented. We do not maintain that subjects see all of a complex instance at the time it is presented. However, when the conditions of presentation are such that they can always re-examine the instance we see no reason for using a special recognition program.

If an instance is shown only once, and then removed, the subject can only store the information which he receives at the time the instance is presented. Here a special "recognition" program is needed. We have a rather primitive method for reading instances into memory in our present model. Included in the initial specification of the artificial subject is a list of dimension sets. Sets are read in the order in which they are placed on the list. During a particular problem our program reads, at every presentation of an instance, all dimension sets which have ever been read. If this provides sufficient information with which to discriminate the current instance from previous instances, the reading process terminates. If sufficient information is not presented, a new dimension set is read, and the discrimination test reapplied. New dimension sets are added until either all dimension sets have been used or discrimination from previously presented instances is possible. When the read program is terminated the appropriate description (some part of the formal description) is entered into internal memory.

For problems in which a requirement for memory exists, a limited occupancy model of human memory is employed (cf. Hunt[8]). The subject parameters specify a certain number of storage cells. These are set aside for representational memory of instances. Each new instance is stored, at random, in one of the cells. The previous content, if any, is lost. Thus, the probability that the artificial subject has a given instance available decreases as the number of intervening instances increases. We consider this model a crude approximation to human memory, although it has been shown to be useful in predicting the probability of utilization of information in certain cases.

Figure 1 represents the very indirect tie between recognition-memory and answer developing-checking units in the present system. It may be that this is not the most effective arrangement. Schemes for joining the subsystems may be considered in a later model.

The "heart" of the model is the answer development subsystem. Its internal procedure is depicted in Figure 2. The answer developing section finds binary decision rules for distinguishing between the denotation of one name and its complement. In doing so it restricts its attention to one dimension set at a time. Dimension sets are selected in the order specified by the current description of the artificial subject. If an answer already exists which involves a particular dimension set, that set will be ignored in answer development. The "executive routine" of the answer developing system is entered when a dimension set is found for which no answer is currently available. The plan followed by the executive routine is to prepare an <u>execution list</u> containing the names of three routines which will be executed in the order specified by the <u>execution list</u>. The contents of internal memory is used as output for the first and second of the three routines, They, in turn, provide the output for the third (last) routine of the execution list. Successful completion of the third routine results in a tentative concept definition.

The executive routine selects routines for the execution list from three reference lists. These are initially specified by the programmer as part of the subject's description list. They may be changed during execution of a simulation.

The first reference list contains the names of <u>ordering routines</u>. Each of these routines splits the instances on internal memory into two sets, working positive and working negative instances. The two categories are mutually exclusive and exhaustive of all instances in memory. In the simulations we have tried thus far three ordering routines are provided. One places in the "working positive" set all instances which are members of a class which has been indicated, by the programmer, as the class for which a concept is to be found. The second currently available routine reverses this procedure, placing the same instances in the working negative set. (If the programmer has indicated that there are several classes of equal importance the class name of the most recently presented instance is used by these two routines.) The third ordering routine defines as "working positives" all those instances which have the class name of the smallest set that is represented in internal memory, provided that there are at least two instances in the set.

Another reference list contains the name of routines which produce a working description of the instances in memory. These routines attach to each instance in internal memory a description based on a transformation of that part of the formal description included in the current dimension set. We have dealt with two description routines. One simply copys the necessary dimensions and values from the formal description to the working description. The other routine defines new dimensions based upon the relation between values of the dimensions of a formal description. The following rules are used to generate the working description:

1. A new dimension is defined for any pair of (source) dimensions whose values are numerical quantities on the same scale. For a particular instance the value of the new dimension is EQUAL, GREATER, or LESS, depending on the comparison of the values of the original pair of dimensions on that instance.

2. A new dimension is defined for any pair of source dimensions on the formal description list if, over the entire set of instances in memory, the two source dimensions share a common value. (The common value need not appear on both dimensions in the same instance.) The value of the new dimension is, for a particular instance, either SAME or DIFFERENT, depending on a comparison of the value of the original dimensions on the instance in question.

In actual programming, the ordering and description routines are applied serially. They are functionally parallel, the output of one does not affect the output of the other. They both provide output to the solution routine. This consists of all instances in internal memory, re-categorized and re-described. The solution routine attempts to define a method for discriminating between working positive and working negative instances. The discrimination is always stated as a definition of the working positive instances, even though these may be members of the complement of the class which the program is trying to define a concept.

At present the model contains three solution routines. The first two are suited for handling conjunctive concept learning problems (problems in which the answer can be stated using only the logical connective <u>and</u>). The third is a conditional procedure which is slower, more complex, and of greater generality.

The two "conjunctive" routines both, as their first operation, list those dimensions which have only one value over the entire set of working positive instances. If this list does not exist no conjunctive definition of the working positive instances exists. If the list does exist, it is handled somewhat differently by the two routines. The first conjunctive routine searches through each of the dimensions to find if one of them <u>never</u> has the same value on the negative instances as it does on all positive instances. The second routine examines all negative instances to see whether any negative instance has the entire conjunction of dimension-value pairs which are common to all positive instances. The routine returns an answer if no such instance can be found. Thus either routine, when it succeeds, defines a conjunctive concept that can be used for the instances in internal storage.

The third solution routine, the conditional routine, is a recursive function which, if slightly modified, would give the artificial subject the capability of answering any concept learning problem. As it currently stands, it provides the capability of solving disjunctive concept learning problems of limited complexity.

The conditional routine first identifies the dimension-value pair which is most frequently found on positive instances. It then generates two sublists of working positives and working negatives, all of which contain this pair. The first conjunctive routine is applied to the two sublists. If it succeeds, it returns with an answer which can be applied to any future instance which has the appropriate dimension-value pair. If it happens that the conditional routine generates only a sublist of positive instances, the answer is the value of the single dimension being considered. If the dimension-value pair does not occur on a future instance, the class membership of this instance is indeterminate.

If an answer is not generated in this manner, or if there remain unclassified instances, the conditional routine is repeated, omitting dimension-value pairs previously considered <u>and</u> any instances which have been classified. The result of the application of the conditional and conjunctive routines constitutes a second "conditional" answer. This procedure is repeated until all instances in internal memory have been classified or until all dimensions have been considered. The result is a classification rule composed of a chain of statements about simple conjunctive answers and the rules under which they apply (e.g. red triangle, green circle). The chain of statements may be of any length, but each statement must contain only two dimension-value pairs. We could have removed this restriction by applying the second conjunctive rule instead of the first. We could also have permitted a nth level conditional rule by applying the conditional routine, recursively, to the sublists until all instances were classified. The resulting procedure would generate a rule for all concept learning problems. It would not necessarily be the most compact statement of the correct rule. It could degenerate into a description of particular instances.

When the executive routine selects an execution list it is, in effect, applying a template for an answer to a particular problem. If the problem has an answer which involves the relevant features abstracted by the ordering and description routines, operating on a particular dimension set, and if the answer is of a particular logical type, there exists an execution list which will find it.

The manner in which our first model changes its template is also indicated in Figure 2. Initially the dimension set is selected. The first execution list is then selected from the reference lists contained in the subject description. The first execution list always uses the routines which are at the top of each reference list. If the execution list cannot obtain an answer, the description or solution routine (alternately) is replaced until the original execution list is re-constructed. When this happens a new ordering routine is selected. The alternation of description and solution routines is repeated until, again, an execution list is repeated. At this point a new ordering routine is selected. When there are no more ordering routines the dimension set is replaced, using the next dimension set on the subject's list of order of noticing dimension sets. The process ends whenever either, an answer is developed, all dimension sets are examined, or, when the allotted time is exceeded. How this is instrumented will be described presently.

During a particular problem the order of dimension sets remains constant. However, during the time when an answer is being developed, the reference lists for description and solution routines may be temporarily altered. This is done by moving a symbol from first to last place on its reference list whenever it is removed from an execution list. One of the ways in which we can simulate individual differences is to change the initial order of routines on the reference lists.

As we have indicated, there is a "time checking" mechanism which may interrupt the answer development process. Associated with each routine on a reference list is an index number. These numbers are specified by the programmer as part of the initial data. The programmer also specifies, as part of the problem data, a number which represents the time that the artificial subject has to develop an answer. Depending on the presentation conditions, this may represent the time he is permitted to spend on the entire problem or the time between stimulus presentations. Every time a routine on an execution list is applied, its index number is subtracted from a time signal which was, originally, set equal to the allowable time number. When the time signal reaches zero answer developing is halted (possibly with the reference lists for description and solution re-arranged) and control is returned from the executive solution routine to a higher level.

The index number associated with each routine can be thought of as an "effort" number, the cost of a particular information processing routine to the subject. Success in any problem depends on a complex interaction between the rules for re-arrangement of order of routines on reference lists, the value of the index number, and the value of the allowable time number. One of our more fascinating research tasks is the unravelling of this relation.

The model, as presently programmed, has an independent check on time. Whenever a new instance is presented it is examined to see if its class membership agrees with that predicted for it by currently active answers. If the new instance does not agree, or (in the case of conditional answers) if no class membership is predicted for that instance, the answer development routine will be entered. If correct prediction occurs the answer development section is entered only if a "slow" rate of stimulus presentation is specified in the problem description.

Whenever an answer is developed the

dimension set and execution list used are stored on its description list. When a problem is solved (i.e. after all instances have been presented), those dimension sets which have been associated with an answer, and those routines which have appeared on successful execution lists, are moved to the head of their respective reference lists. Thus, the characteristics of the subject which were originally specified by the programmer have been modified by the program.

The transfer procedure has an interesting psychological implication. Our artificial subject shows positive or negative transfer only when the preceding problem is solved. Also, transfer is almost entirely dependent upon the form of the immediately preceding problem. We do not know whether or not this is true of human problem solving.

Simulations and Evaluation

The model was not conceived *in vacuo*. Previous, unprogrammed models[7] had been considered for some time. In addition, we gathered protocols from Yale undergraduates who attempted to solve a "concept learning" problem which had three logically correct answers; a disjunction, a conjunction, and a relation. (This problem has been described previously[9] and some data on its difficulty was available.) All three conditions of presentation were given to each subject. The model we have just presented gave the best overall "postdiction" of responses of any model we could devise. In fitting it we altered the order and identity of symbols on reference lists, but otherwise kept the model constant. Since each subject solved three problems, we were able to make some tests of our transfer procedures and thus do not rely too heavily upon pre-specified orders. The results of our match were generally encouraging. However, they cannot be taken as validating evidence since the protocols were used to develop the program.

Some more encouraging evidence came when the artificial subject attempted a series of problems used by Shepard, Hovland and Jenkins[21]. This was a completely separate study. Human subjects were asked to find categorizing rules for each of the six possible types of splits of eight instances, each describable by one of two values on three dimensions, into two sets of four each. Human subjects could solve, quite rapidly, a problem in which all relevant information could be derived from a single dimension. So could our artificial subject. Both human and artificial intelligence found a problem consisting of a "string" of two conditional statements (e.g. big and red, or small and white) easy. In a third case, humans and the artificial subject were unable to develop a workable rule for the authors "Type VI" classification, in which the answer requires either description of each instance or a rather subtle rule about alternation of values. Humans did better than the artificial subject in one situation. When the correct answer could be stated as a simple rule with one exception, our program finds the problem difficult. Humans find it hard, but not nearly as hard as the "Type VI" problem. The results of this simulation, and particularly the discrepance just mentioned, forced us to consider alternate recursions in the conditional solution routine.

A somewhat similar, unpublished, experiment was performed by Hunt and H. H. Wells. Here the five commonly used logical connective between two elements provided the answer. A "truth table" was constructed and presented to subjects in geometric form. For example, the connective "p *and* q" might be represented by "red and star." The five problems were presented in five orders, each subject solving all five problems in one of the orders. Simulation and analysis of this experiment has not been completed at the time of this writing, however, we have some preliminary results. There is good general agreement between our simulation routines and *some* protocols. Both the computer model and the subjects are sensitive to the order in which problems are presented, but their reactions are not as similar as we would like. A new transfer procedure is needed. In an experiment which is not directly related to simulation, Wells is studying the manner in which human subjects learn methods of solution for disjunctive problems. We hope that his experiments will provide some clues about the nature of the transfer procedures we should include in our model.

We do not claim to have presented a complete explanation of concept learning! Certainly others will agree with us. In programming the model we made many decisions with little theoretical or empirical justification. Some of these are certain to be wrong. But which ones?

We shall probably have to change our

routines for memory and recognition. Some of the known phenomenon of memory cannot be reproduced by a simple occupancy model. For instance, the effect of stimulus similarity upon memory cannot be represented. Our model has an all or none aspect in its interference features. An intervening instance either completely eliminates the record of a previous instance or does not affect it at all. This does not seem to be the final answer to the problem of memory in concept learning.

Two alternative memory systems have been considered. One system retains and extends the limited occupancy model. Instead of storing one "codeword" (actually, a list structure), representing all known information about an instance, on a single occupancy list, several code words would be stored in several occupancy lists. Each of these code words would represent a particular type of information about some part of the instance in question. Storage of each codeword would be independent on each occupancy list. Codewords referring to the same instance would reference each other's locations. When information from memory was required a picture of each instance would be reconstructed from the cross referencing system. However, since intervening instances would be storing codewords independently on each occupancy list, some of the codewords might be replaced. The extent of this replacement would depend upon the similarity between the instance to be recalled and the stimuli which followed its presentation. This system would be sensitive to stimulus similarity effects.

Alternately, we could use an associationist memory system. Instead of trying to remember units of information directly we would build "associations" between names and stimulus features. This is the logic of the technique used by many learning theorists in psychology. Machinery to realize such a memory has been extensively investigated by Rosenblatt[17,18]. There is also some similarity between this approach and the classification mechanisms based upon Selfridge's "Pandemonium" scheme[19]. To adopt such a memory system would require changing the entire logic of our model. Association schemes generally contain, in themselves, a mechanism for concept learning. It also seems that they require some sort of gradient of generalization. Recent experiments[20,21] indicate that, in concept learning, the tendency to code stimuli symbolically plays a greater role than generalization based upon stimulus similarity. For these reasons we have, tentatively, rejected an associationist memory mechanism.

In the present model we subject the formal description of an instance to two transformations. When an instance is presented the dimensions of the formal description are sampled to determine what information is to be placed in memory. At some later time, that part of the formal description which is in memory is re-transformed to provide a working description. The two procedures could be combined if the description routine currently at the head of the description routine reference list were to be applied directly to an instance before it entered memory.

Such a procedure would have advantages in saving storage space. Instead of having to have two separate locations, for working and permanent description, in the internal memory, only one description need be stored. But we pay for saving this space by losing information. By definition, any working description can be derived from the formal description. All working descriptions cannot be derived from each other. For instance, if we know that an instance contained two figures of the same color, we do not know what that color is. As a result, our artificial subject's ability to utilize a particular description routine at time $\underline{t}$ would depend very much upon the description routines used previously.

The role of "set" at time of presentation as a determinant of later memory characteristics needs more extensive investigation. Some experiments[12,13] suggest that "set" is a function of how memory is searched rather than how items enter into memory. Also, there exists a rather contradictory literature on "latent learning", a term used to describe experiments in which animals, trained to respond to cue A in the presence of cue B, which is irrelevant to the animal's current need, learn more rapidly a later response to cue B. From present experimental results it is not obvious how stimulus recognition and answer development procedures should be connected in a concept learning simulation.

Procedures for representing transfer may not be represented adequately in the present model. Transfer is defined as the effect of previous problem solving experience upon solution of the problem with which the subject is faced at the time of the test. We decided to work first with a simple method of representing transfer, in which the subject tries whatever worked last time. A principal result of the simulation of the Hunt and Wells work on logical connectives has been a demonstration that a new transfer procedure is needed.

In the tradition of classical learning theory, we could attach a modifiable numerical index to each routine on a reference list. This index could be used to determine the probability that a routine would be selected. This method of representing learning is probably the most common. The principal objection to it is that it implies the existence of "counters in the head" and, essentially, makes our program a digital simulation of an analog system.

The alternative to association indices is a new method of ordinal rearrangement of routines on a reference list. The problem with ordinal re-arrangements is that they do not permit us to specify a variable distance between routines on a list. Suppose we consider each concept learning problem as a contest between routines on the same reference list. The one that finds a place on a successful execution list is victorious. How many times must the routine in position n "win" before it gains the next highest position? Should it jump positions? As we have indicated, some research relevant to this topic is being conducted.

Conceivably, we may have to change our entire method of transfer. At present our model records answers, with associated information about useful routines. We could attach to routines information about problems on which they had been useful. We would then have to develop some way for the artificial subject to extract, rapidly, key features of a problem while the answer is being developed. Routines would be examined to see what, in the light of past experience, was their probable usefulness on this sort of problem.

Closely related to the problem of transfer is the problem of response selection during learning. Our present model rearranges its order of response selection after a problem is solved. During a problem, response selection is controlled by time parameters which are independent of program control. No use is made of intermediate computations in selecting the next item to be placed on an execution list. In an alternate model this might be the controlling factor. The means-end analysis of the Logic Theorist[15] uses intermediate calculations heavily. Amarel[1] has proposed a computer model for an area very similar to ours in which intermediate computations exert control on answer development.

Our simulation work, and analysis of experimental data, has convinced us that some method of making the choice of one item on an execution list dependent upon the product of execution of previously selected routines is desirable. What is not clear is the optimum amount of dependency. Bartlett[2] has presented an analogue, in an entirely different context, which may clarify the problem. He compared problem solving and thinking to motor skills responses, such as serving in tennis. There are certain points at which a chain of responses can be altered, in between these points a series of acts will be executed without interruption. Our problem, experimentally, is to identify the responses and choice points.

We feel that the principal use of our model, so far, has not been in the generating of an explanation of concept learning so much as it has been in indicating the type of new experimental data needed. We have had to be very specific in our thoughts as we programmed this model. As a result, we have reached some conclusions about the kind of experiments that need to be done. It may well be that the typical concept learning experiment confuses three processes; memory, recognition, and symbolic problem solving. It is not clear whether or not these should be treated as part of a unitary "concept learning" act. They can be programmed separately. In addition, we have become concerned with questions of transfer, the effect of the subject's current hypothesis upon his later retention of information, and the effect of time pressure upon information processing. A real awareness of these problems has been a major outcome of programming a concept learning model.

Comparisons with Related Work

Viewed formally, our problem is closely related to models of pattern recognition. Programming either a pattern recognizer or a concept learner involves the development of a mechanism which operates on a specified stimulus universe to map stimuli from predetermined subsets into particular responses. Because of this mathematical identity, at least one critic[10] has suggested that problems of this sort should be treated together, without "psychologizing" or "neurologizing." While this may be useful in developing theorems about a canonical form of categorization, it may not be an appropriate strategy for simulation studies. In particular, our approach is quite different from that of the pattern recognition studies with which we are familiar.

The most striking difference is in the manner in which we pre-code the stimuli. Pattern recognizers usually accept stimuli coded into projections on a grid. The result is a string of bits, each bit representing the presence or absence of illumination of some part of the grid. The same representation could be used for a temporal pattern. Each bit would stand for the presence or absence of some stimulus feature.

We presuppose the existence of a dimension and value coding[6] and deal with perceptual aspects which are readily verbalizable. A pattern recognizer develops its own code. Any coding scheme developed by a pattern recognizer will be specific to the stimuli used (visual vs. auditory, etc.). Since we are interested in the manipulation of coded elements we avoid this problem by fiat in our programming and by explicit instructions to our subjects in our experimental work.

Our model is also different from most pattern recognizers in the processes it uses. Pattern recognizers, at least as developed by Selfridge and his co-workers[19], and by Rosenblatt[17,18], are basically parallel processing devices which utilize a large number of redundant, error prone tests. Our program is a serial processor which tries to develop a single, perhaps complex, error free classification test. We do not see any incompatibility in the two approaches. Pattern recognizers are inspired by the capability of biological systems to amplify upon their sensory inputs. Our program deals with the simulation of a symbolic process. That the two problems are formally similar does not mean that they are realized in the same way by problem solvers.

In principle, there would be no objection to utilizing a pattern recognizer to provide the input to the concept learner. The combined system could develop its own dimensions and values and then operate upon them. In practice, such a scheme is undoubtedly premature. But it is a long range goal.

The concept learning problem has been attacked directly in two previously mentioned studies by Kochen[11] and Amarel[1]. Kochen restricted his program to solution of "concepts" based upon a conjunctive rule involving stimuli specified by strings of bits. His program consisted of executing algorithms upon the information about the universe of objects which was available at any one time, in memory. The program also contained features for making random guesses about the correct concept. These guesses could be weighed for "confidence", using an index which satisfied Polya's[16] postulates for plausible reasoning. One of Kochen's findings, based on Monte Carlo runs of his system, was that changes in the value of the confidence index could be used to estimate the probability that an answer was correct before a proof of the answer was available.

Amarel[1] proposed a machine that could generate routines to map arguments to values in symbolic logic. The key feature of his proposal, one we might well adopt, is his use of intermediate results to "monitor" future answer development.

Neither Kochen nor Amarel were directly concerned with simulation of human performance. This difference in goals, and features of programming, are the major differences between our work and theirs.

Superficially, our program is similar to the list processing programs written by the Carnegie Institute of Technology-RAND Corporation group headed by Newell, Shaw, and Simon, and McCarthy[14] and his associates at M.I.T. In particular, the work of Feigenbaum[4], at Carnegie, is related to ours. He developed a program to simulate paired-associates learning. As part of his program he included a routine for selective recognition of stimulus features. As more experience with the stimulus universe was provided, more features were read into the system to enable it to make finer discriminations. The logic of Feigenbaum's recognizing system, and in particular its capability for dropping stimulus features which are not useful in discrimination, could be incorporated into our program.

Our present program, although running now, is in no sense complete. Almost every new simulation has indicated ways in which it could be improved. We intend to continue to investigate concept learning by use of an information processing model. But we do wish to add a word of caution.

Neither our model, nor any other, has generated a large number of new experiments. This is a traditional test of the utility of a scientific model, and it is going to have to be met by us and by others interested in this field. We do not feel that the utility of computer programming models in psychology has been proven or disproven. The jury is still out. We, of course, hope that a favorable verdict will be returned.

References

1. Amarel, S. An approach to automatic theory formation. Paper presented at the Illinois Symposium on the Principles of Self Organization, 1960.

2. Bartlett, F. C. Thinking. New York: Basic Books, 1958.

3. Church, A. Introduction to mathematical logic, vol. I. Princeton, N.J.: Princeton U. Press, 1956.

4. Feigenbaum, E. An information processing theory of verbal learning. RAND Corp. publication, p. 1817, 1959.

5. Green, B. F. IPL-V, the Newell-Shaw-Simon programming language. Behavioral Science, 1960, 5, #1.

6. Hovland, C. I. A "communication analysis" of concept learning. Psychol. Rev., 1952, 59, 461-472.

7. Hovland, C. I. and Hunt, E. B. The computer simulation of concept attainment. Behavioral Science, 1960, 5, 265-267.

8. Hunt, E. B. An experimental analysis and computer simulation of the role of memory in concept learning. Unpubl. Ph.D. dissertation, Yale U., 1960.

9. Hunt, E. B. and Hovland, C. I. Order of consideration of different types of concepts. J. exp. Psychol., 1960, 59, 220-225.

10. Keller, H. Finite automata, pattern recognition, and perceptrons. AEC Computing Center and Applied Mathematics Center, Report NYO-2884, 1960.

11. Kochen, M. Experimental study of hypothesis formulation by computer. IBM Research report, RC-294. IBM Research Center, Yorktown Heights, New York, 1960.

12. Lawrence, D. H. and Coles, G. R. Accuracy of recognition with alternatives before and after the stimulus. J. exp. Psychol., 1954, 47, 208-214.

13. Lawrence, D. H. and LaBerge, D. L. The relationship between recognition accuracy and order of reporting stimulus dimensions. J. exp. Psychol., 1956, 51, 12-18.

14. McCarthy, J. Recursive functions of symbolic expressions and their computation by machine. Communications of the Association for Computing Machinery, April, 1960.

15. Newell, A. and Shaw, J. C. Programming the logic theory machine. RAND Corp. publication, p. 954, 1957.

16. Polya, G. Mathematics and plausible reasoning. Princeton: Princeton U. Press, 1954.

17. Rosenblatt, F. The perceptron, a probabilistic model for information organization and storage in the brain. Psychol. Rev., 1958, 65, 368-408.

18. Rosenblatt, F. Perceptual generalization over transformation groups. In Self organizing systems, London, Pergamon Press, 1959.

19. Selfridge, O. and Neisser, U. Pattern recognition. Scientific American, 1960, 203, 60-79.

20. Shepard, R. N. and Chang, J. J. Stimulus generalization in the learning of classifications. Bell Telephone Lab. mimeographed report, 1961.

21. Shepard, R. N., Hovland, C. I. and Jenkins, H. M. Learning and memorization of classifications. Psychol. Monogr., 1961, in press.

Acknowledgment

The work reported in this paper was supported by a grant for the study of concept learning, from the Ford Foundation to Carl I. Hovland. The computational work involved was supported, for the most part, by the Computation Center, Massachusetts Institute of Technology. Dr. Bert F. Green, Jr., and Alice Wolf, of Lincoln Laboratory, M.I.T. made available the 709 version of the IPL-V interpreter and instructed the first author in its use. The aid received is greatfully acknowledged.

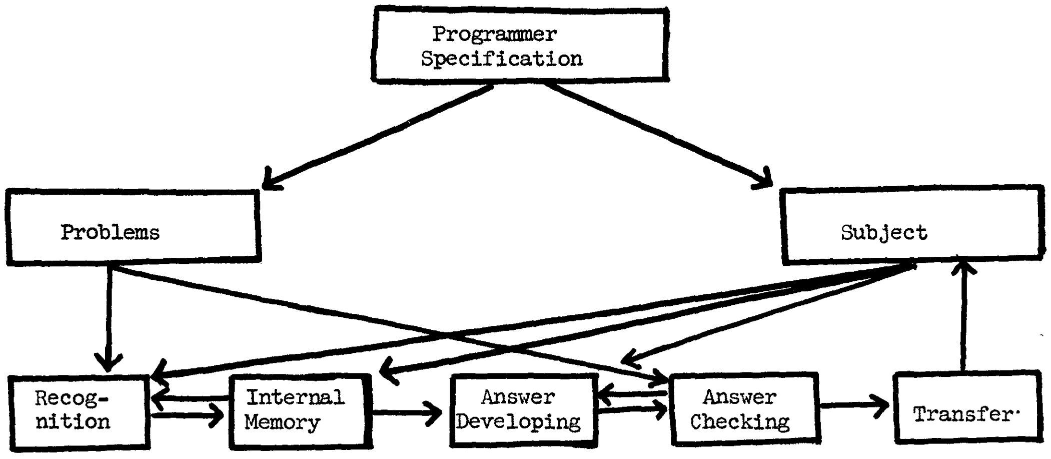


Figure 1. Program Control Chart.

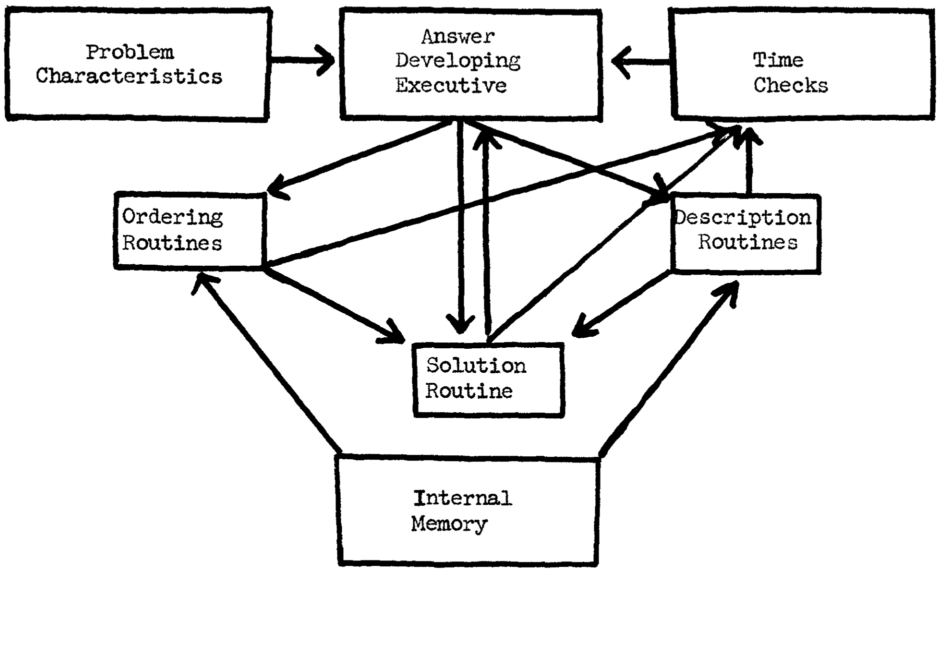


Figure 2. Answer Developing Procedure.

PARALLELISM IN COMPUTER ORGANIZATION RANDOM NUMBER GENERATION IN THE FIXED-PLUS-VARIABLE COMPUTER SYSTEM

M. Aoki*, G. Estrin*, and T. Tang**
Department of Engineering
University of California*
National Cash Register Co.**
Los Angeles, California

Summary

The Fixed-Plus-Variable Structure Computer System[1] utilizes an inventory of modules which can be interconnected as special purpose configurations operating simultaneously with other parts of the system. Since the structure considered makes no permanent committment of hardware to relatively rarely used operations it permits reconsideration of designs previously discarded as uneconomic. Problem formulations utilizing random numbers generally require large numbers of trials to achieve high confidence results. Despite the fact that in most problems random number generation does not require a large percentage of the total computing time, it may be desirable to eliminate even that time by use of special purpose random number generators.

Special purpose circuits can be designed to generate pseudo-random numbers in parallel with other activities such that these numbers are available on demand in the same sense as any other operand stored for use in the computation.

This paper discusses a number of different methods of generating pseudo-random numbers, the time required in existing programs, the hardware implications of different parallel and serial designs. The criteria for choice of one method over another in the context of particular problems and the Fixed-Plus-Variable Computer System are evaluated.

1. Introduction

The Fixed-Plus-Variable Structure Computer System[1] is organized such that there is associated with a general purpose computer an inventory of modules which may be reorganized as needed to reduce the overall computation time required for solution of particular problems. The special purpose configurations established in the variable structure part of the system may carry on their operations simultaneously with those of the fixed part of the system and may have their configurations altered during the course of a given problem.

In general, of course, any of the operations which may be executed by the special purpose configuration may either be programmed in the fixed structure general purpose part of the system or may be added to the list of permanently available commands. The latter course will not in general be taken for relatively rarely used operations and both methods imply sequential operation on all commercially available systems. Since in basic concept the (F + V) system[1] makes no permanent committment of its variable structure inventory, it is reasonable to consider its application to many problems where special purpose techniques would previously have been discarded as uneconomic.

This paper considers the class of problems which makes use of long sequences of random numbers for simulation of complex processes using Monte Carlo methods.[2-4] In most such problems the operations which must be done on the random number are more time consuming than the random generations themselves and deserve most of the attention of special purpose equipment. However it will be found that the large number of operations generally will make it worth while to utilize some fraction of the variable structure inventory to essentially remove the time for random number generation from the overall computation time.

2. Methods for Generating and Testing Random Numbers

2.1 Introduction

The successful use of the Monte Carlo method in digital computers[32] depends on* a good supply of long sequences of random numbers. There are, broadly speaking, four ways of supplying such sequences.

The preparation of this paper was sponsored, in part, by the Office of Naval Research and the Atomic Energy Commission. Reproduction in whole or part is permitted for any purpose of the United States Government.

* During the course of the calculations associated with any Monte Carlo problem, it is also necessary to produce random variables according to a variety of different probability distributions. A common method of doing this is to incorporate into the computing machine a sequence of uniformly distributed random numbers. The desired random variables are then obtained by transformations based on them[11].
 There are three methods which can be adapted to serve this purpose. These are based on direct functional transformations, the principle of compound probabilities, and the procedure of rejecting part of the sampled values according to an appropriate test or rule.[5,6] The first is called the "direct method", the second the "composition method" and the third the "rejection method".

1. Tables of random numbers may be recorded on paper tape, punched cards or magnetic tape. The tapes or cards are fed into the computer at suitable stages of the calculation. This method has found little favor in those problems which require very long sequences of random numbers since the time taken to read in the tables may soon become excessive. For example the IBM 711 card reader operates at the rate of 250 cards per minute or 6,000 words per minute.[7] If a problem requires one million random numbers, it will take the card reader 166.66 minutes to transfer the data into the computer. If magnetic tape is used, over 100 seconds would be required on the fastest available magnetic tape system.

2. Random numbers may be generated by physical processes such as radioactivity or discharges in gases. The chief objection is a rather paradoxical one; the number sequences cannot be repeated and so it is very difficult to check the calculation because it is not always possible to distinguish between variations in results due to random fluctuations and those due to changes in the program or even to the faulty running of the computer.

3. Pseudo-random numbers may be generated by arithmetical processes on general purpose high speed digital computers. The most common methods are the mid-square process and additive and multiplicative conqruence methods.[8,9]

Whenever the time required for generation of pseudo-random numbers in a problem is significant, a special purpose computer or wired-in instruction may be considered utilizing any of the methods in (3). Such a special purpose configuration would essentially deliver a pseudo-random number on demand and might operate in parallel with any other processes going on.

There are three criteria which should be satisfied by any method of generation of pseudo-random numbers.

1. The numbers generated should satisfy the tests of randomness prescribed by the user.
2. The rate of generation of pseudo-random numbers should compare favorably with the rate at which they may be used in typical computations.
3. The recursion relation should produce a sufficiently long sequence of pseudo-random numbers. All sequences generated by arithmetic process are either cyclic or enter a cycle if prolonged far enough. A cyclic sequence will not be a satisfactory source of pseudo-random numbers unless the period is so great that it is of no consequence in practical computations.

2.2 The Testing of Sequences for Randomness

The tests of Kendall and Smith[10] have been used frequently on sequences of decimal digits. They may be adopted in various ways for testing sequences of binary digits generated by pseudo-random number generators. These tests are the frequency test, the serial test, the poker test and runs test. In all these tests the deviations of the observed counts from those expected from a perfectly random sequence are studied. Chi-squared tests are usually used to give a measure of the permissible deviations from expectations.[11]

In practical applications of various tests of significance, the 5%, 1% and 0.1% levels of significance are often used. Some empirical results related to the four tests below can be found in papers by Green[8] and Rotenberg[9].

1. The frequency test. The primary requirement for randomness is that the numbers, considered as fixed point, positive fractions, be uniformly distributed over their range $(0 \leq X \leq 1)$

To test this, the binary numbers can be separated into octal digits, and a tally of the frequency of occurrence of each octal numeral can be made and tested since a random process should produce uniformity in every octal digit position.

2. Serial correlations. Another vital requirement is that the successive generated numbers be independent of others.

The correlation coefficient ρ of two one-dimensional random variables X and Y is defined by

$$\rho = \frac{\mu_{11}}{\sqrt{\mu_{20}\ \mu_{02}}} \qquad (2.1)$$

$$\text{where } \mu_{ij} = E\left[(X-E(X))^i \ (Y-E(Y))^j\right].$$

If the two random variable are independent we have $\mu_{11}=0$ and thus $\rho=0$. Thus two independent random variables are always uncorrelated, although the converse is not always true.[11]

One can compute from the generated sequence of pseudo-random numbers the ρ between the successive terms and between two numbers k positions apart, k>1, by defining X and Y appropriately.

If the computed ρ is far from zero one rejects the hypothesis that the sequence of pseudo-random numbers are independent, although one has no guarantee of their independence even if the computed ρ is close to zero.

3. The poker test. The mutual independence of the various digit positions can also be checked by the poker test. For each generated number, a designated five octal digits are treated as a poker hand, without regard to suit, and the hand is tallied as five of a kind, four of a kind, full house, three of a kind, two pairs, one pair, or bust. Table 1 shows the expected frequency of poker hands.

Table 1 [12]

Class	Symbol	Expected frequency
Busts	abcde	302.4
Pairs	aabcd	504.0
Two Pairs	aabbc	108.0
Threes	aaabc	72.0
Full house	aaabb	9.0
Fours	aaaab	4.5
Fives	aaaaa	0.1
		1,000.0

Good pseudo-random numbers should not deviate significantly from these expected frequencies.[13]

4. Runs test.[13] Another aspect of serial dependency can be checked by finding the distribution of runs above and below some constant and runs up and down as described below. If the observations are randomly drawn from the same population we do not expect very long runs of either type, and the occurrence of such runs will, therefore, usually be taken as indicating non-randomness.

a. Runs above and below the median.[13,14] Consider a random arrangement of n elements consisting of n_1 a's and n_2 b's, $n = n_1 + n_2$, such as the following arrangement of 11 a's and 14 b's:

babaaaababbbaababbbbbabba

An uninterrupted sequence of elements of the same kind is called a run, and the length of a run is given by the number of elements defining the run. The sequence above begins with a run of one b, then follows a run of one a and so on.

Assuming that all possible arrangements occur with the same probability we may find the distribution function of the number of runs of a's of length i by means of combinatorial theory.

For small values of n the distribution function of the mean number of runs of a's of length i, R_{1i}, and R_{2i}, similarly defined for runs of b's, may be tabulated by writing down all possible arrangements of a's and b's, as shown in Table 2 for n=6 and $n_1 = n_2 = 3$. The number of different arrangements is $\binom{6}{3} = 20$. The expected number of runs of both kinds of elements of length k or more is approximately given by

$$\frac{n}{2^k} \qquad (2.2)$$

from which we get the expected total number of runs, R, as

$$R = \frac{n+2}{2} \qquad (2.3)$$

The theory above may be applied to samples from a population with a continuous distribution function by classifying sample values larger than the sample median as a's and values smaller than the median as b's.

In some cases the population median is known or a hypothetical value is tested. It will be noted that in this case the number of a's will be a random variable, having a binomial distribution with ${}^{n}1/_{n} = 0.5$.

b. Runs up and down.[14] Consider a sequence of n different observations, $X_1, X_2 \ldots , X_n$ and the sequence of signs (+ or –) of the (n-1) differences $X_{i+1} - X_i$. A sequence of successive + signs is called a run up and a sequence of successive – signs is called a run down. The length of a run is given by the number of equal signs defining the run. The total number of runs is denoted by R, the number of runs of length i by r_i and the number of runs of length k or more by ${}^{i}R_k$, $R_k = \sum r_i$, $i \geq k$.

Table 2

No.	Arrangement	R	R_{11}	R_{12}	R_{13}	R_{21}	R_{22}	R_{23}
1	aaabbb	2	1	1	1	1	1	1
2	aabbba	3	2	1	0	1	1	1
3	abbbaa	3	2	1	0	1	1	1
4	aababb	4	2	1	0	2	1	0
5	aabbab	4	2	1	0	2	1	0
6	abaabb	4	2	1	0	2	1	0
7	abbaab	4	2	1	0	2	1	0
8	ababba	5	3	0	0	2	1	0
9	abbaba	5	3	0	0	2	1	0
10	ababab	6	3	0	0	3	0	0
11	bababa	6	3	0	0	3	0	0
12	baabab	5	2	1	0	3	0	0
13	babaab	5	2	1	0	3	0	0
14	baabba	4	2	1	0	2	1	0
15	babbaa	4	2	1	0	2	1	0
16	bbaaba	4	2	1	0	2	1	0
17	bbabaa	4	2	1	0	2	1	0
18	baaabb	3	1	1	1	2	1	0
19	bbaaab	3	1	1	1	2	1	0
20	bbbaaa	2	1	1	1	1	1	1
	Total	80	40	16	4	40	16	4
	Mean	4	2	0.8	0.2	2	0.8	0.2

For example, the sequence of 15 elements: 39,42,38,53,51,30,40,28,43,46,52,55,29,24,34, leads to the following sequence of signs of differences between successive elements:

+ – + — + – + + + + – – +

The sequence is thus characterized by 4 runs up of length 1, one run up of length 4, 2 runs down of length 1, and 2 runs down of length 2, giving a total of 9 runs.

Assuming that all n! possible arrangements of the n numbers occur with the same probability, we have the following mean number of runs.

$$\bar{r}_i = \frac{2}{(i+3)}\left[n\ (i^2 + 3 + 1) - (i^3 + 3_i^{\,2} - i - 4)\right], \quad i \leq n-2 \qquad (2.4)$$

and

$$\bar{R}_K = \frac{3}{(k+2)}\left[n\ (k+1) - (k^2 + k - 1)\right], k \leq n-1 \qquad (2.5)$$

Table 3 shows the expected number of runs up and down of length k or more in random arrangements of n different numbers. The values $\bar{R}_k$ computed from generated pseudo-random numbers should not again differ too much from the theoretical values.

5. Discussion. In a particular application a block of digits of a definite length may be required to be random when considered in isolation. The remark of Kendall and Smith[10] is relevant here, namely "if a series S is locally random in a Domain, it does not follow that any part of S is locally random in that Domain." They conclude that a set of random numbers which is adequate for all requirements is impossible, and the only solution is to carry out tests on blocks of numbers and give the results of these tests, so that the prospective user can choose from the tables these

Table 3

k	$\bar{R}_K$
1	(1/3)(2n-1)
2	(1/12)(3n-5)
3	(1/60)(4n-11)
4	(1/360)(5n-19)
5	(1/2520)(6n-29)
6	(1/20160)(7n-41)
7	(1/181440)(8n-55)

blocks which are suitable for his problem.

2.3 Methods of Generation of Pseudo-Random Sequences

Methods of generation of pseudo-random bers by high speed computers are usually divided into two classes.

a. Those for which there is, or appears to be, no way of determining the cycle structure other than the brute force procedure of investigating the sequence by actual computation. The best known method of this type is the middle-of-square method.

b. Those for which mathematical analysis can determine the cycle structure, and even suggest suitable parameters in the recursion relation to give the longest period and most satisfactory output such as the congruential methods, additive and multiplicative.

1. The Middle of Square Method[15]. Von Neumann and Metropolis first suggested the middle of squares method. This process can be exemplified in a special case as follows. Take a 4 digit number X_o, e.g., $X_o = 2061$. Square it to obtain 04247721. Define

$X_1 = 2477$, the middle four digits of X_o^2, as the pseudo-random number.

Next $X_1^2 = 06135529$ and $X_2 = 1355$ the second pseudo-random number. Similarly $X_3 = 8360$, $X_4 = 8896$, etc.

For the results of some tests on numbers generated this way see Mauchly[16] and Votam and Rafferty[17]. Unsatisfactory results have been observed if the number has less than eight digits and the sequence may develop unsatisfactory properties if extended beyond 700 or so eight-digit numbers.

2. The Congruential methods.

a. The additive process[8]. Starting with an initial set of n random numbers, $X_1, X_2 \ldots, X_n$ in the range

$$0 < X < 1$$

the procedure generates additional numbers, X_j, successively according to the following rule:

$$X_j = (X_{j-1} + X_{j-n}) \text{ Mod } 1, \; j>n \qquad (2.6)$$

An equivalent statement of the process that is convenient for binary machines is

$$X_j = (X_{j-1} + X_{j-n}) \text{ Mod } 2^r \quad j>n \qquad (2.7)$$

where $0 < X_j < 2^r$ and r is the maximum number of bits used to encode each fixed point number.

The reason that equation (2.7) is very convenient for binary machines is that the modulus additions[18] are performed by simply adding and disregarding overflow.

To illustrate the behavior of this type of sequence, consider the following simple example:

if $n = 2$
$X_1 = 0, \quad X_2 = 1, \quad r = 3$

Equation (2.7) becomes

$$X_j = (X_{j-1} - X_{j-n}) \quad \text{Mod } 2^3 \qquad (2.8)$$

Then

$$X_3 = (X_2 - X_1) \text{ Mod } 2^3 = 1$$

$$X_4 = (X_3 - X_2) \text{ Mod } 2^3 = 2$$

By the same procedure, the following sequence is obtained

0, 1, 1, 2, 3, 5, 0, 5, 5, 2, 7, 1

Then the sequence will repeat itself.

Length of Period. As just mentioned, the additive process is periodic; it will eventually repeat itself by generating the original n numbers. The period depends mainly on n and r. In Green, Smith and Klem's paper,[8] they state that the period T

$$T = K_n 2^{r-1} \qquad (2.9)$$

where K_n is a constant depending on n and is given in Table 4

Table 4

n	K_n	n	K_n	n	K_n
2	3	7	127	12	3255
3	7	8	63	13	2905
4	15	9	73	14	11811
5	21	10	889	15	32767
6	63	11	2047	16	255

For n = 15 and r = 35 the period is

$$T = 32767 \times 2^{34}$$

which is approximately equal to 5×10^{14}.

Four statistical tests[10] were made by Green, Smith and Klem of the apparent randomness of numbers generated by the additive process of equation (2.7) with various n and r.

The additive process passes the frequency, poker, and serial correlation tests, but it fails the runs test for n less than 16. It passes the runs test for n = 16, and presumably for n greater than 16. If alternate numbers are discarded, then the additive process passes the runs test for n = 6, and presumably for all larger n.

One wishing to use the additive process to generate random numbers has three alternatives. He may decide that he needs only approximate randomness, as in setting up experimental designs, and may therefore ignore the runs difficulty. He will then choose n according to his convenience. The runs test is very sensitive and failure to pass this test does not mean that the numbers are badly awry.

However, if one is working on a Monte Carlo problem, or some similar problem with stringent randomness requirements, then he must either choose an n of at least 16, or must plan to discard alternate number.

b. The multiplicative method. Although the additive process for generating random numbers has been found very convenient for use in digital computers, there exists an even simpler process which was first suggested by Lehmer[19] and later by Greenberger[20] and Rotenberg[9].

$$X_{i+1} = (2^a + 1)\,X_i + C, \text{ Mod } 2^p \qquad (2.10)$$

The advantages of this process over the additive process are as follows:

(a) The process does not require an initial set of random numbers. This means that no memory storage is needed.

(b) Multiplication by a power of the base can be accomplished by shifting, which is comparable in speed to addition.

(c) This process requires essentially three additions and it can be done in one logical step by a special purpose digital computer.

(d) This process also permits construction of a special serial computer with very few components.

Several empirical tests were made by Rothenberg of the apparent randomness of the numbers generated by the process of Equation (2.10) with a = 7, C = 1 and p = 35. These tests are: the frequency test, the runs up and down test and the runs above and below the mean test. The results show that the numbers are uniformly distributed and that there is no serial correlation in the sequence.

The serial correlation coefficient between two consecutive numbers of this sequence is shown by Coveyou[21],

$$\rho(X_i, X_{i+1}) = \frac{1-6\,\frac{C}{P}(1-\frac{C}{P})}{2^a + 1} \qquad (2.11)$$

where P is the modulus of X_i, i.e. $P = 2^p$,

for a = 7 , C = 1

$$\rho = \frac{1}{2^7 + 1} = 0.008$$

By taking a = 9, this correlation coefficient can be reduced to approximately 0.002.

It can be shown[21] that the sequence of Equation (2.10) can generate the full period of 2^p numbers if

$$a \geq 2 \text{ and C is ODD .}$$

See appendix I for the proof.

3. Designs of a Special Purpose Random Number Generator

As a result of the increased interest in the use of Monte Carlo methods of computation in high speed digital computers a number of subroutines have been written.

For example, an IBM 709 program for the additive process

$$x_j = (x_{j-1} + x_{j-n}), \text{ Mod } 2^{35} \qquad (3.1)$$

requires 11 instructions and twenty-two 709 machine cycles, i.e., 264 microseconds to generate one number. See appendix II for code.

An IBM 709 program for the multiplicative process

$$x_j = (2^a + 1)\,X_{j-1} + C, \text{ Mod } 2^{35} \qquad (3.2)$$

requires 7 instructions and fourteen 709 machine cycles, i.e., 168 microseconds to generate one pseudo-random number. The actual code is listed in appendix II.

A Monte Carlo problem which demands one million random numbers is not considered to be impracticable. Notice that this does not mean that the problem has undergone one million computational trials. There are usually many random variables involved in one trial run. To generate one million random numbers by the multiplicative method requires 168 seconds on the 709.

The random number generation time can be reduced, if a fast machine is used. The 7090 would take 33.6 microseconds to generate one number by using the multiplicative process, which is five times faster than the 709. However, the percentage of the generation time consumed in solving the problem is not going to be changed. For example, in the matrix inversion problem it is 17% of the total problem running time no matter what computer is being used. And in some other problems, it could be even more significant than this 17%.

In the paper "Organization of Computer System The Fixed Plus Variable Structure Computer" by Estrin[1], it is pointed out that "... The fastest single component switching speeds being discussed are about 10^{-9} second. For any significant parallel information transfers most researchers observe a loss of a factor of 10^2 or 10^3 bringing basic parallel computer operations to 10^{-6} or 10^{-7} second.." One cannot expect to do very much better with the populations of switching elements without a change in the method of organizing them... The primary goal of the Fixed Plus Variable Structure Computer is:

"1. To permit computations which are beyond the capabilities of present systems by providing an inventory of high speed substructures and rules for interconnecting them such that the entire system may be temporarily distorted into a problem oriented special purpose computer......"

Special purpose circuits can be designed to generate random numbers in parallel with other activities in the computer such that these numbers are available on demand in the same sense as any other operand stored for use in the computation.

The special purpose generator can be some fraction of the variable structure inventory in the (F + V) system to essentially remove the time for random number generation from the overall computation time.

In the following sections, the design of a serial and a parallel random number generator will be discussed. The principal criterion for the serial generator is the use of a small number of components while maintaining a reasonable speed; for the parallel generator, the achievement of high speed with a reasonable number of components.

The serial generator would increase the performance in solving many Monte Carlo problems. It is possible to generate one random number every eight microseconds continuously.

The parallel generator may achieve an access time of less than 0.14 microseconds. Although the parallel generator does not appear to have immediate usefulness it may be compatible with future super high speed computers and particular problems formulated for solution by Monte Carlo techniques.

4. Parallel and Serial Designs of a Special Purpose Random Number Generator - Utilizing A Multiplicative Congruence Method

4.1 Functional Description

This pseudo-random number generator is essentially a special purpose computer, which will produce long sequences of uniformly distributed random numbers. The algorithm used is based on the Multiplicative Congruential method,

$$x_j = (2^a + 1)\, x_{j-1} + C, \text{ Mod } 2^p \qquad (4.3)$$

As indicated in section 3 , the choice of the parameters a, c, and p is governed by the period desired and the minimum acceptable value of the serial correlation coefficient.

In this example the constants are chosen in such form as to illustrate the computer design and may be modified according to the above criteria .

The constants of the above equation are chosen as follows:*

$a = 11$

$C = 1$

$p = 20$

The length of the period is equal to 2^{20} which is approximately one million. For a = 11, the serial correlation coefficient is approximately equal to 2^{-11}.

The generator can be set to any initial starting number. Since the process does not require storage of the previous number, the generator will go on to generate the next random number as soon as the previous one has been transferred.

1. The Serial Generator. The Functional Block Diagram is shown in Figure 1. The complete functional flow includes two cycles:

Cycle 1: $(2^{11} + 1)\, x_i$ is formed

Cycle 2: $X_{j+1} = (2^{11} + 1)\, x_i + 1$ is completed.

The R register is a 20-bit register in which R_1 initially contains the least significant bit and which stores the random number, X_i. In every logical sequential step, the 20-bit register is shifted right one place. The output of the adder is shifted into the most significant bit of the R register, R_{20}, and the least significant bit of the R register is shifted into the I register.

The T register controls the cycles during the flow. T = 0 corresponds to Cycle 1. T = 1 corresponds to Cycle 2.

The N register counts from 0 to 19 and provides the required sequence of 20 control steps.

The S register controls the start signal. It also provides appropriate control signals to all other registers in the generator. The S register is set by an initial "Demand" signal from a supervisory control observing the needs of the rest of the (F + V) system and is reset at the end of Cycle 2.

a. Input equations. The input equations may be written by collecting together and combining the timing and truth table logic as follows:

$$S = (\text{Demand})S' + (N1 + N2 + N3 + N4 + N5)\, S + T$$

$$N1 = S\,[N1']$$

$$N2 = S\,[N1N2' + N1'N2]$$

$$N3 = S\,[N1N2N3'N5' + (N1' + N2')\; N3]$$

$$N4 = S\,[N1N2N3N4' + (N1' + N2' + N3')\, N4]$$

$$N5 = S\,[N1N2N3N4N5' + (N1' + N2')\, N5]$$

$$T = T'N5N2N1 + T\,(N5' + N2' + N1')$$

* If $C \neq 1$, then C must be added into the proper position of a parallel adder or at an appropriate time in a serial adder rather than the simple forced carry into the least significant position.

$R_n = R_{n+1}$ $(n = 2, 3, \ldots\ldots\ldots, 19)$

$J_n = I1\ I2'C_{n-1}' + I1'\ I2\ C_{n-1}' + I1'\ I2'C_{n-1}$

$+ I1\ I2\ C_{n-1}$

where J_n is the adder output

$I1 = R1$

$I2 + R10\ T'(N5 + N4\ N3 + N4\ N2\ N1)$

$C_n = I1\ I2 + C_{n-1}\ (I1 + I2) + T'\ N5\ N2\ N1$

b. Estimation of running time for the full cycles. If five megacycle flip-flops are used, the delay through each flip-flop is 200 millimicroseconds. It takes forty-clock times to complete a cycle. The total delay, therefore, is eight microseconds.

2. The Parallel Generator. The functional block diagram is shown in Figure 2.

The R register stores the random number x_i which is 20 bits long. The two inputs to the adder are x_i and $2^{11}x_i$. The constant C = 1, is added to the sum x_i and $2^{11}x_i$ by setting the initial carry. As soon as the Start signal is received, the adder outputs will transfer x_{i+1} into R register to replace x_i. Since x_i, $2^{11}x_i$ and C are all available at the same time, the process can be accomplished in one logical step. For cases where $C \neq 1$ more complete stages of the adder may be required to permit injection of C. The access time of this special computer will mainly depend upon the speed of the adder.

A parallel adder must be able to generate parallel carry functions.[22,23] Since carry C_k is an explicit function of C_{k-1}, the parallel carry functions can only be obtained by a method of substitution. In applying this method, one will soon find out that the carry functions contain a great number of terms. This may make it electronically impossible to mechanize and in any event the response of the carry functions becomes so slow that it actually loses the effectiveness of a parallel adder.

If an adder of the type proposed by Weinberger and Smith[22] is used, the logical configuration of Figure 3 is obtained.

By studying the adder equations, one finds that the maximum fan-in is four. The maximum fan-out is five. This results in a maximum of six inverter delay times and one flip-flop delay time. If one uses as a basic building block, a current mode, diode gate unsaturated inverter with a gain-bandwidth product of approximately 400 megacycles, then one may use inverters having a response time of less than 15 millimicroseconds per stage. The total delay through the adder is less than 90 mμs. If 20 megacycle flip-flops are used, the delay through each flip-flop is 50 mμs. Under such conditions it is implied that the random number generator could generate a random number every 140 mμs.

4.2 (F + V) Structure

Given any sets of functions which are to be solved, it is always possible to design a special purpose computer which can perform the operation faster and more efficiently than a general purpose computer can. It is clear, however, that the speed and simplicity of the special purpose computer has been achieved at a complete sacrifice of flexibility. The lack of flexibility, in general, means that the control unit for a special purpose computer can be simplified since the sequence of operations performed is fixed. The data to be operated on can be arranged so that it becomes available as required without the necessity for addresses or even a memory cycle.

A general purpose computer is usually more expensive than any one special purpose computer. However, aside from the fields of automatic control and business data processing, one will hardly find the extended use of a special purpose computer. A general purpose computer is mostly used as an arithmetic tool, which can be programmed to solve thousands of functions separately and continuously. It is possible but more likely impractical to have a great number of fixed special purpose computers working simultaneously together. Either it is a tremendous waste in the dollar point of view or the overall performance will probably be disappointing due to the difficulty of data synchronization among machines.

In order to take advantage of the better technology of both types of computers; that is, the speed and simple organization of the special purpose computer and the flexibility of the general purpose computer and still be within the realism of electronic circuitry knowhow, the proposed "Fixed Plus Variable Structure Computer" is advantageous. The variable structure inventory in the (F + V) system would be expandable and could grow with expandable and could grow with experiences and needs. The procedures which lead to the design of special purpose modules from the variable structure inventory can be set up in standard forms such as the SHARE program for a general purpose computer system.

The special purpose random number generator will be considered as one of the special modules in the (F + V) system. It should be noticed that the particular serial random number generator discussed in the previous section consists of only five conventional switching elements. They are:

1. A shift register
2. A three input serial full adder
3. A counter
4. Three flip-flops
5. Three "and" gates and three "or" gates

The parallel generator consists of the following two parts:

1. A register
2. A high speed parallel adder

It is also suggested in the (F + V) system that if the random number generator is used frequently enough, it should become part of the instructionset of a future general purpose computer or even possibly become one of the modules in the V inventory. By adding this special purpose random number generator to any presently exisitng general purpose computer, for solving Monte Carlo problems, it becomes a minimum (F + V) system at the expense of a fraction of the original cost but achieves a significant gain in speed.

5. Examples of Applications

5.1 Introduction

In this section, three problems using Monte Carlo technique, i.e., gamma ray diffusion problem, matrix inversion problem and design of electronic curcuits, are considered. The role which is played by sequences of pseudo-random numbers will be emphasized as well as the possibility of simultaneous generations of other elementary functions such as logarithmic and exponential functions in the (F+V) structure computer.

5.2 Gamma ray diffusion[24,25]

In the Monte Carlo approach to the Gamma ray diffusion problem, a beam of monoenergetic Gamma rays is incident at a given angle on a plane parallel barrier of finite thickness in one dimension and of infinite length in the other two dimensions.

The paths of the Gamma rays are simulated by appropriate random walks. A set of three random numbers is generated per random walk between successive events such as collision or absorption in order to obtain energy and angular distribution of transmitted and reflected Gamma rays.

In this problem it turns out that simultaneous computations of elementary functions ln x and e^x can reduce the total computation time by 40%. The generation of pseudo-random numbers requires 7.4% of the total computing time.[24,25,36,37,38,39]

5.3 Matrix Inversion

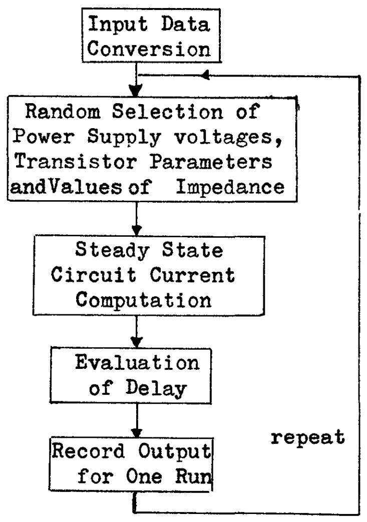
Matrix inversion by the Monte Carlo method was first suggested by Von Neumann and Ulam and later described by Forsythe and Liebler[26]. The method provides a simple computational approach to the statistical estimation of the elements of the inverse of a given matrix. Here, random walks are again introduced in generating the matrix inverse and the random walks are terminated when the statistical variation of the result is less than a given tolerance. The percentage of time spent in generating random numbers for this type of program is approximately 17.3%.[26,27,28]

5.4 Electronic Circuit Design[31,33,34,35]

The Monte Carlo technique is finding wider application in designing electronic circuits and is illustrated by the problem of estimating propagation delay time of transistor-resistor logic circuits.[31]

An important consideration in the design of a transistor resistor logic system is the delay in propagating signals through various levels of these circuits. The propagation delay is an implicit non-linear function of many variables. In investigation of its statistical properties, two immediate difficulties are encountered. First, the distributions of the variables are not completely known. While it is possible to make reasonable assumptions regarding resistor values, such as a uniform density function, the transistor parameters present a different picture. Their density functions are not generally known in a form describable by any simple function such as the normal distribution. One approach to this problem is to derive approximations to these functions from empirical measurement. Even with the distribution of all the variables known, the distribution of the resultant delay cannot be readily determined and the techniques of the Monte Carlo method become very powerful aids in the solution of such complex problems. This is particularly true since the development of models for the transient and steady state analysis of transistor switching circuits such as those of Ebers and Moll[29] and Sparks and Beaufoy.[30]

The flow chart below shows a typical computational flow for a conventional sequential machine in applying the Monte Carlo technique to evaluate the propagation delay.


Input Data Conversion
Random Selection of Power Supply voltages, Transistor Parameters and Values of Impedance
Steady State Circuit Current Computation
Evaluation of Delay
Record Output for One Run
repeat

The method of approach is: establish a mathematical model of the circuit; generate a set of numerical values for all these variables according to their respective distributions;* evaluate the function using this set of values; and repeat the above operations using another independent set of data.

* One can obtain these values by actually measuring the parameters of a particular transistor randomly selected. Resistor values may be assumed to have a uniform distribution.

Calculation of Rise Time, T_o

The time required for the collector current to reach 90% of its final value is

$$T_o = \tau_a \frac{I_{b1}}{I_{b1} - \frac{0.9 I_c}{\beta_n}} \qquad (5.1)$$

where four random numbers are generated to select

τ_a	: minority carrier lifetime in the active region
I_{b1}	: step turn on current
I_c	: collector current
β_n	: normal common emitter current gain

Calculation of Storage Time — T_1

The time required to switch the transistor from the saturation region to the edge of the active region is

$$T_1 = \tau_s \ln \frac{I_{b2} - I_{b1}}{I_{b2} - \frac{I_c}{\beta_n}} \qquad (5.2)$$

where two new random numbers are needed to obtain

τ_s	: minority carrier lifetime in the saturation region
I_{b2}	: step turn off current

Calculation of Decay Time T_2

The time required for the collector current to decay to zero is

$$T_2 = \tau_a \ln \frac{\frac{I_c}{\beta_n} - I_{b2}}{\frac{0.1\, I_c}{\beta_n} - I_{b2}} \qquad (5.3)$$

Calculation of the Input Delay time T_{OD}

The time required to charge or discharge the input capacitance is

$$T_{OD} = RC_{in} \ln \frac{e^{\frac{T_2}{RC_{in}}} - 1}{\frac{T_2}{RC_{in}}} \quad \frac{V_2 - V_1}{V_2} \qquad (5.4)$$

where at least four additional random numbers are used to select

R	: the effective impedance looking out from the base terminal
C_{in}	: input capacitance
V_2	: forward bias voltage on base
V_1	: reverse bias voltage on base

In estimating R, it may be more desirable to randomly select individual resistor values in the circuit and then compute R, rather than picking R randomly from some estimated distribution for R.

Turn on

$$T_{on} = T_o + \tau_a \frac{T_{OD}}{T_2} + \frac{e^{\frac{1}{\tau_a (T_2 - T_{OD})}} - 1}{\frac{T_2}{\tau_a}} \qquad (5.5)$$

In expression (5.2), T_o is the response of the transistor to a step of base current. The second term compensates the delay due to the input capacitance.

Storage Time

$$T_s = T_1 + \tau_s \ln \frac{e^{\frac{T_{on}}{\tau_s}} - 1}{\frac{T_{on}}{\tau_s}} \qquad (5.6)$$

In expression (5.6) T_1 is the time required to remove excess carriers from the base with a step of base current applied. The second term in expression (5.6) is due to the fact that the base of the transistor is discharged by a signal with a finite rise time, T_{on}.

A simulation program has been prepared by Y. C. Ho and W. J. Dunnett.[31] The entire program has approximately 4,000 instructions and takes an average of three seconds per run on the IBM 709 Computer. In order to verify and establish the accuracy of the mathematical model a statistical experiment was also performed. In this experiment measured data was compared with predicted statistical data provided by the computer program and in each case, the predicted mean was quite close to the measured mean. Moreover, the computed deviation in every case was more conservative than its measured value.

One notices that in computing the propagation delay, logarithmic and exponential functions are used four times and three times respectively, in addition to the required generation of a set of ten random numbers per transistor per run.

The special purpose random number generator does away with most of the time required to generate random numbers which become available essentially in one access time to the high speed memory. If in the same sense as the random number generation, a special purpose configuration is established in V for computation of ln x or e^x concurrently with the program being executed in F, then one may expect to achieve significant reduction in total computation time.

Consideration of the set of equations 5.1-5.6 shows that the seven natural logrithm and exponential functions must be evaluated in sequence. Thus while F is calculating the operands to be used in these functions, V might be generating the random numbers and functions required in the following steps.

A block diagram of the (F + V) structure computer illustrating a form for achieving the above is shown in Figure 4.

6. Conclusion

In all three examples considered in Section 5, serial generation of random numbers appeared quite satisfactory, since, using inexpensive transistors it is possible to generate one random number in about 8 μs. No case has been found in which it is necessary to employ the parallel method of random number generation.

If it becomes necessary to generate random numbers more rapidly than 8 μs per random number, one can either decide to employ more expensive, faster components in a serial random number generator or utilize the fully parallel configuration.

References

1. Estrin, G, "Organization of Computer Systems - The Fixed Plus Variable Structure Computer," Proceedings of Western Joint Computer Conference, May, 1960.

2. Donsker, M. D. and Kac, M., "The Monte Carlo Method and Its Applications", Proceedings of Computation Seminar, IBM Corp., December, 1949.

3. McCracken, D. D., "The Monte Carlo Method", Scientific American, Vol. 192, Page 90, May, 1955.

4. Brown, G. W. "Monte Carlo Methods", Modern Mathematics for the Engineer, Edited by E. Beckenbach, McGraw-Hill Co., New York, 1956.

5. Lytle, E. J., "A Description of the Generation and Testing of a Set of Random Normal Deviates", Symposium on Monte Carlo Methods, Wiley, 1956.

6. Butler, J. W., "Machine Sampling from Given Probability Distributions", Symposium on Monte Carlo Methods, Wiley, 1956.

7. IBM 709 Data Processing System Reference Manual, 1958.

8. Green, B. F., Smith J. E. K., and Klem, L., "Empirical Tests of an Additive Random Number Generator", Journal of the Assoc. Comp. Machinery, pp. 527-537, September, 1959.

9. Rotenberg, A., "A New Pseudo-Random Number Generator", Journal of the Assoc. for Comp. Machinery, pp. 75-77, January, 1960.

10. Kendall, M. G. and Smith, B. B., "Randomness and Random Sampling Numbers", Journal of the Royal Statistical Society, Vol. 101, pp. 147-166, 1938.

11. Cramer, H., Mathematical Methods of Statistics, Princeton University Press, 1946.

12. Rand Corp., A Million Random Digits With 100,000 Normal Deviates., Free Press, 1955.

13. Dixon, W. J. and Massey, F. J., Introduction to Statistical Analysis, McGraw-Hill, 1951.

14. Hald, A., Statistical Theory with Engineering Applications, Wiley, New York, 1952.

15. Taussky, O. and Todd, J., "Generation of Pseudo Random Numbers", Symposium on Monte Carlo Methods, Wiley, 1956.

16. Mauchly, J. W., Pseudo-Random Numbers, Presented to American Statistical Assoc., December 29, 1949.

17. Votan, D. F. and Rafferty, J. A., "High Speed Sampling", MTAC 5 (1951) pp. 1-8.

18. Garner, Harvey L., "The Residue Number System", Proceedings of Electronic Computers 8 June, 1959.

19. Lehmer, D. H., "Mathematical Methods in Large Scale Computing Units," Proc. of a 2nd Symp. on Large-Scale Digital Calculating Machinery, pp. 141-146, 1949.

20. Greenberger, M., Decision Unit Models and Simulation of the United States Economy, Chapter III, (Preliminary Draft 1958).

21. Coveyou, R. R., "Serial Correlation in the Generation of Pseudo-Random Numbers," J. Assoc. Comp., pp. 72-74, January, 1960.

22. Weinberger, A. and Smith, J., "A One-Microsecond Adder Using One Microsecond Circuitry," Proceedings of Electronic Computers, IRE EC-5, June, 1956.

23. Richards, R. K., Arithmatic Operations in Digital Computers, D. Van Nostrand, 1955.

24. Berger, M. J., "An Application of the Monte Carlo Method to a Problem in Gamma Ray Diffusion," Symposium on Monte Carlo Methods, Wiley, 1956.

25. Beach, L. A. and Theus, R. B., "Stochastic

Calculations of Gamma Ray Diffusion, Symposium on Monte Carlo Methods, Wiley, 1956.

26. Forsythe, G. E. and Leibler, R. A., "Matrix Inversion by a Monte Carlo Method," MTAC IV 31: pp. 127-129, 1950.

27. Tang, T. "A Special Purpose Random Number Generator" Master Thesis, U.C.L.A., January, 1960.

28. Ralston, A. and Wilf, H. S. Mathematical Method for Digital Computers, John Wiley and Sons, Inc., New York, 1960.

29. Ebers, J. J. and Moll, J. L. "Large-Signal Behavior of Junction Transistors," Proceedings of IRE, December, 1954.

30. Beaufoy, R. and Sparks, J. J. "The Junction Transistor as a Charge Controlled Device," A.T.E. Journal, October, 1957.

31. Ho, Y. C. and W. J. Dunnett, "Monte Carlo Analysis of Transistor Resistor Logic Circuit," IRE Convention Record, March, 1960.

32. Edmonds, A. R., "The Generation of Pseudo-Random Numbers On Electronic Digital Computers," The Computer Journal, pp. 181-185, January, 1960.

33. Hellerman, L. and Racite, M. P., "Reliability Techniques for Electronic Circuit Design," Transaction on Reliability and Quality Control, IRE September, 1958.

34. Parker, J. B., "The Accumulation of Chance Effects and the Gaussian Frequency Distribution," Phil Mag., 38:681-682 (1947)

35. Silberstein, L. "The Accumulation of Chance Effects and the Gaussian Frequency Distribution," Phil Mag., 35:395-404 (1944).

36. IBM Share Program F114 Distribution No. 027, Western Data Processing Center, U.C.L.A.

37. IBM Share Program LAQ1 Distribution No. 525, Western Data Processing Center, U.C.L.A.

38. IBM Share Program AS09 Distribution No. 224, Western Data Processing Center, U.C.L.A.

39. IBM Share Program AS03 Distribution No. 224, Western Data Processing Center, U.C.L.A.

Appendix I

Period of Multiplicative Congruence Method

From the equation

$$X_{i+1} = (2^a + 1)\, X_i + C, \text{ Mod } 2^P \qquad (1)$$

and

$$\begin{aligned} X_{i+2} &= (2^a + 1)\, X_{i+1} + C \\ &= (2^a + 1)\left[(2^a+1)\, X_i + C \right] + C \\ &= (2^a + 1)^2 X_i + (2^a + 1)\, C + C \end{aligned}$$

and

$$\begin{aligned} X_{i+n} &= (2^a + 1)^n X_i + \left[(2^a + 1)^{n-1} + (2^a + 1)^{n-2} + \dots \right] C \qquad (2) \\ &= (2^{a+1})^n X_i + C \, \frac{(2^a + 1)^n - 1}{2^a} \end{aligned}$$

The sequence repeats itself when

$$X_{i+n} = X_i \qquad (3)$$

From (2) and (3) we have

$$\frac{(2^a+1)^n - 1}{2^a} \left[2^a X_i + C \right] = 0 \quad \text{Mod } 2^P \qquad (4)$$

If C is odd and $a \geq 2$ the sum of the terms in the bracket is odd. Then the first term must be divisible by 2^P in order to satisfy (3). Thus it is required to find the smallest n satisfying the congruence

$$\frac{(2^a + 1)^n - 1}{2^a} = 0 \quad \text{Mod } 2^P \qquad (4)$$

In Rotenberg's paper[9], he points out that from the Number Theory, $\phi\,(2^P)$ is a solution of (4) and the smallest n must be of the form 2^r.

Expanding Equation (4) by the binomial theorem we have

$$\begin{aligned} \frac{(1 + 2^a)^{2^r} - 1}{2^a} &= \frac{1}{2^a} \left[1 + 2^r 2^a + \frac{2^r (2^r - 1) 2^{2a}}{2^1} + \dots - 1 \right] \\ &= \frac{2^r 2^a}{2^a} \left[1 + \frac{2^a (2^r - 1)}{2} + \dots \right] \\ &= 2^r \left[1 + 2^{a-z} (2^{r-1}) + \dots \right] \end{aligned}$$

If a = 2, all terms in the bracket after the first "1" will be even and thus the whole bracket is odd. Then to satisfy (4) the factor 2^r must be divisible by 2^P. Thus the minimum value for r is P, or $n = 2^P$.

Appendix II

Examples of Computer Routines to Generate Random Numbers

The IBM 709 Additive Congruence Method[14]

Location	Operation	Address Tag	Comments
00	SXD	Rand 1, 1	The contents of index reg. 1 is stored in Rand 1
01	LXD	Rand 2, 1	No. of initial numbers is loaded into index reg. 1
02	CLA	Rand 3, 1	x_{j-1} is added to acc.
04	TIX	05, 1, 0	Transfer on index (10)
05	ADD	Rand 3, 1	x_{j-n} is added to x_{j-1}
06	STO	Rand 3, 1	x_j is stored
07	SXD	Rand 2, 1	The content of index reg. 1 is stored in Rand 2
08	LXD	Rand 2, 1	Index reg. 1 is restored
09	TOV	10	overflow condition reset
10	TRA	Normal Return	

Rand 1 Temporary Storage
Rand 2 No. of initial numbers
Rand 3 Address of the last initial random number

(Random numbers are stored in consecutive memory locations)

The two SXD and LXD (00, 01, 07, 08) instructions could be omitted if the index register 1 were available for use by the subroutine, thus saving four instructions per generated number.

The IBM 709 Multiplicative Congruence Method

Location	Operation	Address	Comments
0	CLA	Rand 1	x_{j-1} is added to acc.
1	ADD	Rand 2	C is added to x_{j-1}
2	STO	Rand 1	x_{j-1}+ C is stored
3	ALS	Rand 3	$2^a x_{j-1}$ is formed
4	ADD	Rand 1	x_j is formed
5	STO	Rand 1	x_j is stored
6	TOV	Normal Return	

Rand 1 Random number x_{j-1}

Rand 2 C

Rand 3 a

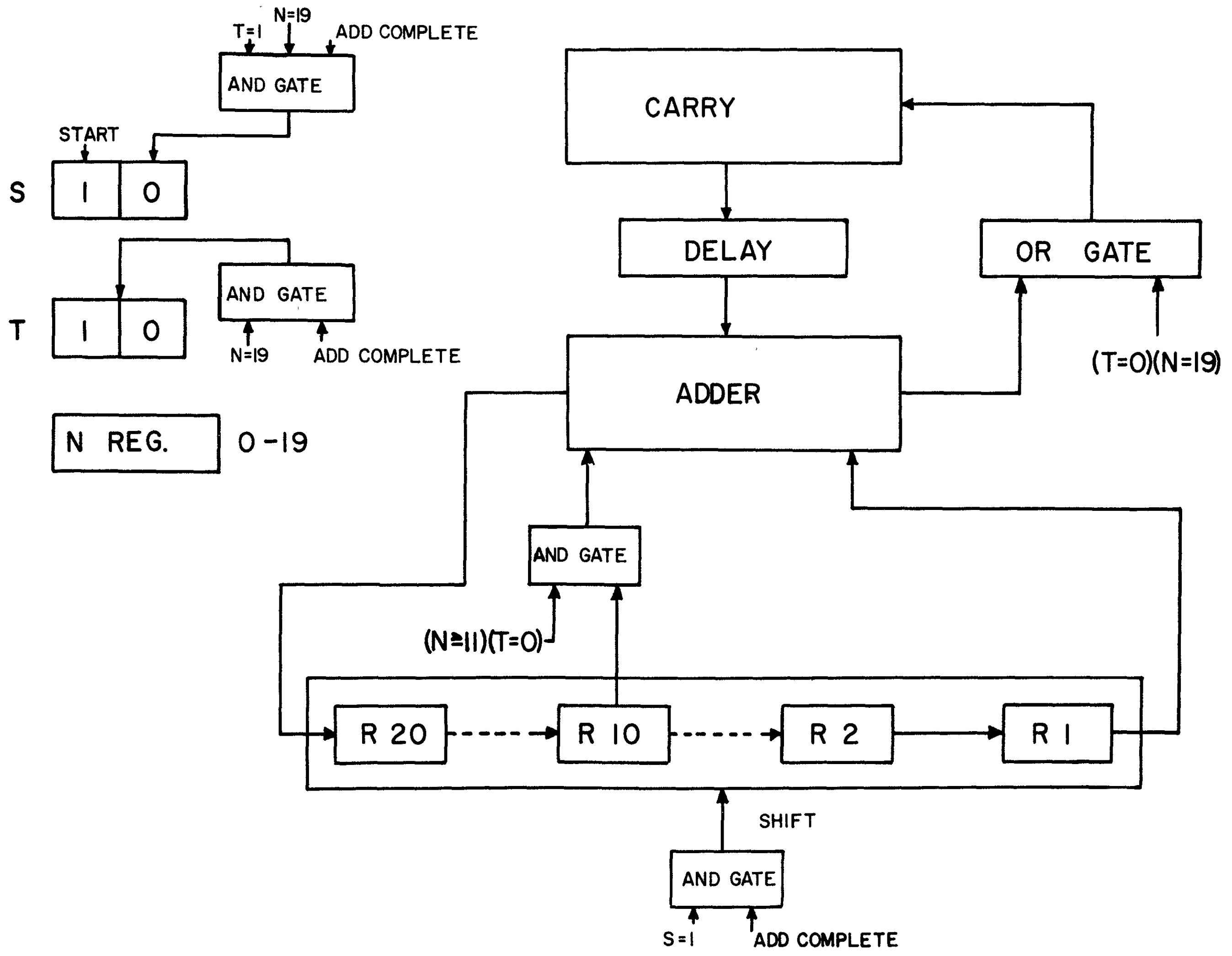


Figure 1 FUNCTIONAL BLOCK DIAGRAM FOR A SERIAL PSEUDO-RANDOM NUMBER GENERATOR

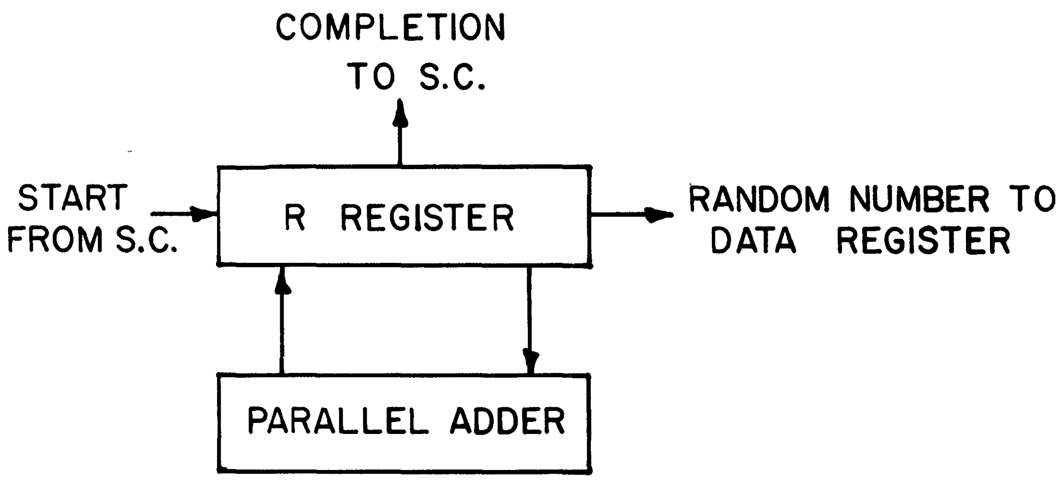


Figure 2 FUNCTIONAL BLOCK DIAGRAM FOR A PARALLEL PSEUDO-RANDOM NUMBER GENERATOR

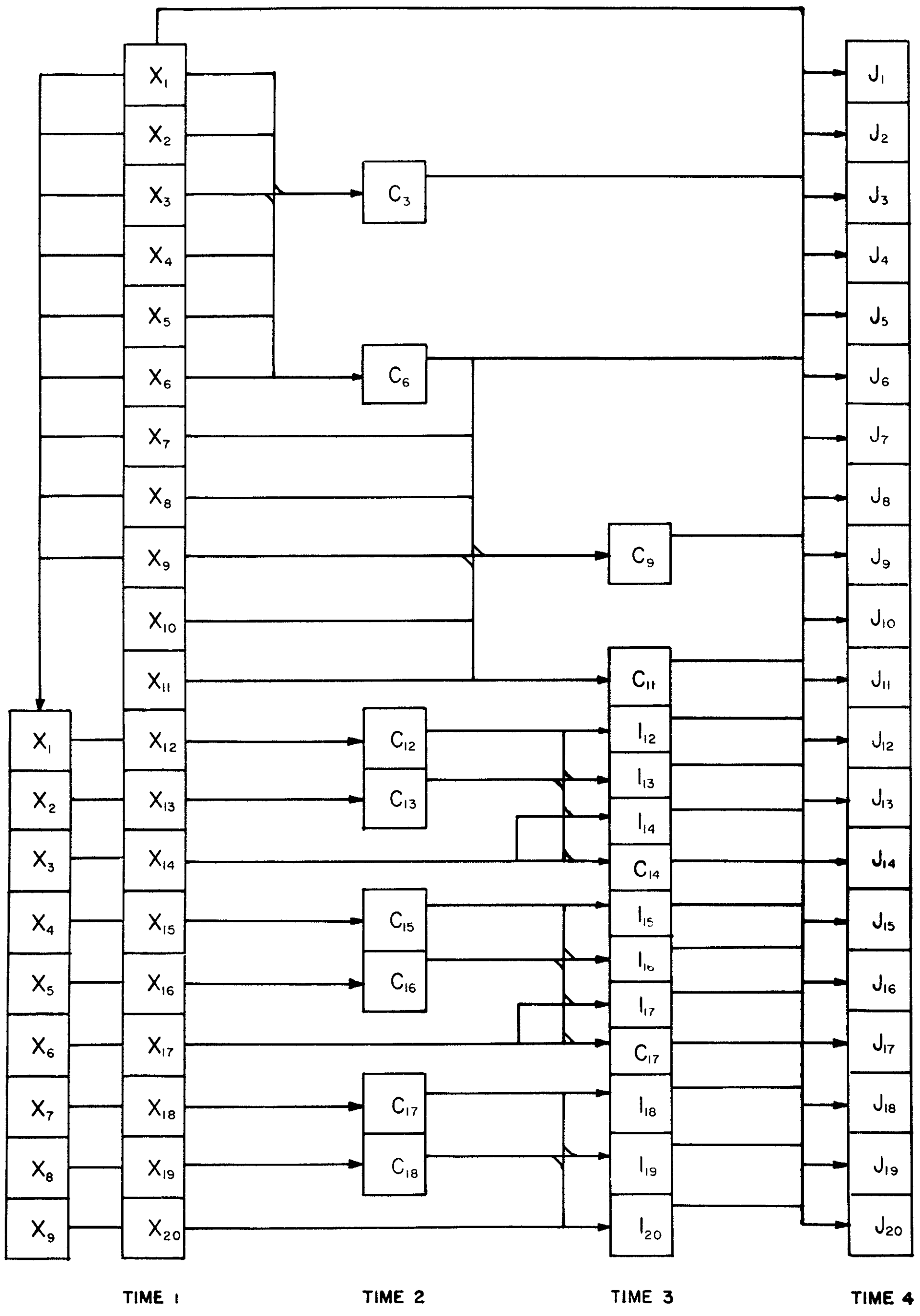


Figure 3 SYMBOLIC FLOW CHART FOR PARALLEL ADDER INDICATING TIMING DURING ADDITION

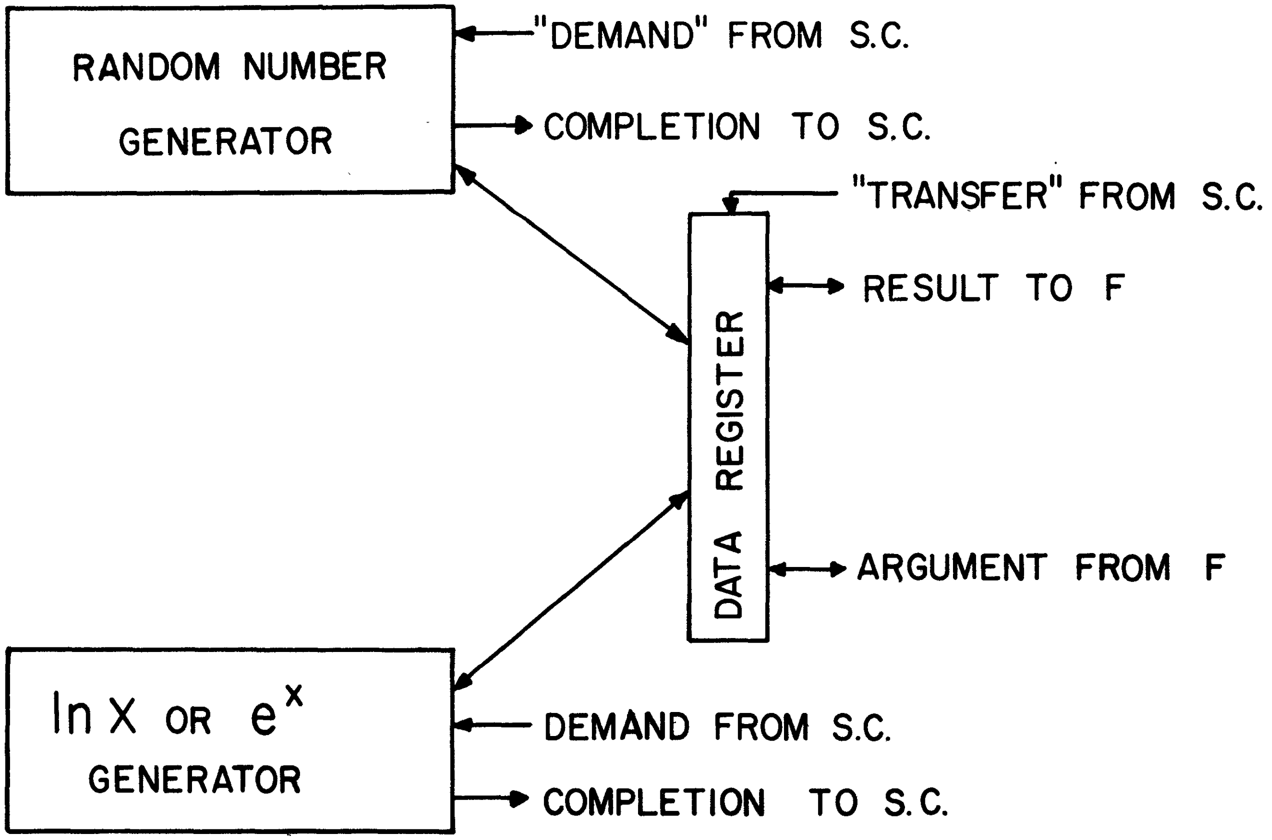


Figure 4 (F + V) STRUCTURE CONFIGURATION

THE CELLSCAN T.M. SYSTEM - A LEUCOCYTE PATTERN ANALYZER

K. Preston, Jr.
The Perkin-Elmer Corporation
Norwalk, Connecticut

Summary

Medical workers suspect that the incidence of the binucleate lymphocyte in the peripheral blood stream is an index of radiation damage.[1] The fact that the incidence of this type of white cell is of the order of one per each million blood cells makes practical use of this index by the human observer essentially impossible. The Atomic Energy Commission has therefore arranged for the construction of a blood cell scanning system with which to determine the feasibility of the semi-automatic identification of the binucleate lymphocyte in glass-mounted smears of human blood.

The CELLSCAN system consists of a closed circuit television microscope coupled to a special purpose digital computer. The system produces a quantized image of the leucocyte whose pattern is to be analyzed. The computer program causes the quantized image to be operated upon serially in such a way that groupings of contiguous binary "1's" are reduced to single 1's. Sampling the number of isolated 1's at various instants during the reduction process produces a histogram of the leucocyte's various constituent parts. The CELLSCAN system is intended to determine whether the resultant histogram of the binucleate lymphocyte is unique in the histogram population of all leucocytes.

Introduction

The use of radioactive materials or of radiation producing equipment continues to increase throughout the world. Users are confronted with the problem of measuring the effect of radiation on workers in radiation environments. To date no good index has been found which measures the physiological changes produced in humans by low level radiation. One hypothesis that has been advanced is that the appearance of a rare type of white blood cell in the blood stream may provide such an index. The rate of occurrence of this white cell is extremely low. In order to obtain sufficient statistical data to prove the above hypotheses, it is essential that automatic equipment be developed which is capable of scanning and analyzing blood samples.

It is the purpose of this paper to describe an experimental blood cell scanning system which has been constructed for the Atomic Energy Commission. The equipment which has been assembled is designed to scan individual white cells rather than blood samples containing a multitude of cells. Experimental results will be used to establish the feasibility of using electro-optical scanning with data processing pattern recognizing techniques to identify certain types of human blood cells.

Hematological Background

Figure 1 is a table showing some of the classes of cells which are found in the human blood stream, i.e. the peripheral blood. Blood cells are divided into two major classes: The red cells (erythrocytes) and the white cells (leucocytes). It is the white cell population with which we are concerned here. The two major classes of white cells are lymphocytes and granulocytes. There are about two granulocytes for each lymphocyte. The sub-class of lymphocytes whose incidence may be an index of low level radiation damage consists of lymphocytes having two nuclei. These cells are called binucleate lymphocytes. For brevity we shall refer to them as "bilobes".

Some quantitative information on the incidence of bilobes has been obtained in experiments with animals. Figure 2 shows a graph prepared by Dr. Marylou Ingram at the University of Rochester.[1] It shows the number of bilobes per thousand lymphocytes in the peripheral blood of dogs who were exposed to low level radiation from the University of Rochester cyclotron. As can be seen the normal incidence of bilobes in the dogs tested is about one per 30,000 lymphocytes. There is an immediate increase in the number of bilobes upon exposure by about an order of magnitude. This increased incidence continues in the peripheral blood for approximately one month during which time its amplitude gradually diminishes.

The normal incidence of bilobes in man is somewhat greater than that given in Figure 2 for dogs. It is of the order of one per ten-thousand white cells which is equivalent to about one for each one million total blood cells. In order to measure the index of bilobes for a particular human being, the technician employed to process a blood sample must count and catalogue cells for several hours. This makes routine measurements of this index both practically and economically unattractive as long as human technicians are required to perform the measurements. These measurements are also subject to error to technician fatigue. For this reason a study program has been instituted in order to determine whether it is within the present state-of-the-art to evolve a blood cell scanning and data processing system which will identify bilobes in typical samples of peripheral human blood with reasonable error rates. This system is called the CELLSCAN system.

Figure 3 shows a portion of a typical blood sample. The sample is prepared by squeezing a drop of human blood between two glass microscope coverslips. When the slips are separated the surface of each is coated with a monocellular layer of blood. The layer of blood is then treated with certain biological dyes which selectively stain the different cells and cell constituents. The chemical reaction between the dye and the cells imparts characteristic colors to the cells which assist the hematologist in cell identification. Figure 3, however, is purposely black and white as the CELLSCAN system uses a monochromatic input.

Different types of white cells appear in Figure 3 interspersed with red cells. The red cells are the regular disc-shaped objects. The white cells are the irregularly shaped darker objects. Each white cell consists of nuclei within the cell body or "cytoplasm". A bilobe is shown as well as two granulocytes. Granulocytes are characterized by the multitude of dark grains or "granules" within their cytoplasm. It should be noted that one granulocyte has two nuclei. Therefore, in order to identify the bilobe, the CELLSCAN system must differentiate between granulocytes and lymphocytes as well as between binucleate lymphocytes and lymphocytes with other than two nuclei.

The technique used by the CELLSCAN system to differentiate between classes of white cells is to count and size the constituents of a given cell. By "constituents" we mean the nuclei and the granules. The output of the system is thus a histogram from which cell type may be determined. A digital approach to this problem of particle counting and sizing has been chosen as the best method for dealing with irregularly shaped particles.

The "Shrink" Algorithm

The equipment which has been designed to identify bilobes employs a television microscanner to deliver a binary video image to a special purpose computer. The scanner is so designed that, whenever white cell nuclei or granules appear in the field of view, the digital output is "1". All other areas of the field of view are "0". A hypothetical binary image is shown in Figure 4a. Since the computer is digital, this image is quantized into a Cartesian array of elements or "bits". Dark elements in Figure 4 correspond to binary "1's".

The function performed by the special purpose computer is to count and size the image areas which are comprised of contiguous "1's". The output of the computer is an image histogram for the particular cell which is viewed by the scanner. In order to produce this histogram the computer first stores the entire binary image. It then operates sequentially on each image bit causing binary "1's" on the periphery of any contiguous group of "1's" to be changed to "0's". This operation has been called the "shrink" operation and was originally suggested by M. Golay.[2] In general, the computer is required to make several sequential examinations of the image before the image histogram is completed. Each examination will be called a "pass". Figure 4b indicates how the shrink operation modifies the original image shown in Figure 4a after two complete passes. Figure 4c indicates the residual image after sufficient passes have been made to reduce the original image to a single isolated binary "1". The number of passes required to reduce the original image to an isolated "1" is proportional to the maximum chord of the original image. Thus the image histogram is computed by counting the number of isolated ones present in memory after each pass. This information is then converted into a plot of the total number of groups of "1's" in the original image which fall into each of several maximum chord ranges.

To program the computer to perform the shrink operation an algorithm has been devised by L. Scott and R. M. Landsman.[2] The algebraic expressions which define the shrink algorithm are shown in Figure 5. The image bit operated upon is designated as X and its neighbors as A, B, C, ...,H. Since the computer sequences from left to right and top to bottom, bits A, B, C, and H have previously been processed. Their processed values are designated A_p, B_p, C_p, and H_p.

Three functions are derived from the values of these 8 neighbors. The function f(ISO) indicates whether the bit under examination is an isolated one while the function f(TAZ) indicates whether the neighbors contain three adjacent "0's". A further function is required whose value indicates whether the X bit is a link between subgroups of a given grouping of contiguous "1's". This indication is provided by the value of f(TUP) which is "1" when there are three or more unlike neighbor pairs. This is necessary so as to prevent the computer program from producing two isolated "1's" when operating upon a dumbbell shaped original image.

The complete shrink algorithm is $f(X_p)$ and is given in the last line of Figure 5. When this algorithm is applied to a stored image, results as previously described in Figure 4 are obtained. It causes isolated "1's" to be retained rather than being converted to "0". This permits periodic sampling of the number of isolated "1's" in the computer memory. This periodic sampling takes place at intervals determined by the scale required for the image histogram.

Technical Details

The first model of the CELLSCAN System has been built in order to evaluate the feasibility of recognizing bilobes using the data processing technique described above. A block diagram is shown in Figure 6. The first model is not intended to be a machine capable of scanning complete blood samples. The micro-scanner is manually centered on the blood cell to be analyzed. For reasons of economy the scanning and data processing rates are slow. About 10 minutes are required to process one image. The periodic computer sampling of isolated "1's" is converted to an image histogram by manual calculation.

The Scanner

The scanner standard employs a Leitz Ortholux microscope coupled to a Dage Data-Vision scanner which has been modified so as to enhance its signal to noise ratio. The output of the video amplifier is digitized by means of a video quantizing circuit.

The Data-Vision equipment scans at a rate of 60 horizontal scans per second which is about 300 times slower than standard closed circuit television systems. The period of the vertical scan is 5 seconds. This produces a field of 300 horizontal scans for each image. This slow scanning rate was chosen so that the output data rate of the scanner could be matched to the data processing rate of the computer. It permits image data recording directly upon magnetic tape at 2000 bits per second. Thus a relatively inexpensive audio tape machine can be employed.

White cells are typically 10 to 20 microns in diameter. In order to restrict the field of view to a single cell an area of blood sample 30 microns square is imaged by the micro-scanner. Thus, with 300 scanning lines, the elementary sample area or scanner "resolution" is one-tenth of a micron square. This resolution is required since the separation between nuclei in a bilobe may be of this order. It implies, however, that computer storage of almost 100,000 bits per field of view is required. In order to reduce this memory requirement a processing technique developed by W. K. Taylor is used in the scanner.[3] Circuitry is used which stretches white-polarity video signals and equally shrinks dark-polarity video signals. This causes the separation between bilobe nuclei to appear greater. However, it simultaneously makes granules appear smaller. In the CELLSCAN system a five fold stretching is permissible. Although granules having maximum horizontal chords less than one-half micron disappear from the image, sufficient granules remain to make cell identification possible. In this fashion the horizontal resolution is decreased to one-half a micron and the computer memory capacity requirement becomes about 20,000 bits.

The Computer

A block diagram of the CELLSCAN system computer is shown in Figure 7. The computer memory is a continuously moving loop of magnetic tape which is capable of storing 19,200 bits corresponding to a 64 x 300 field of view. Non-return-to-zero recording is used with automatic erase after reading. One track is used for storing "0's"; another, for "1's". Internal computer timing signals are obtained from the data stored on the tape. At the beginning of the data processing run the computer memory is erased. Then under control of synchronizing signals from the scanner the memory is filled with one field of view. The image stored on the magnetic tape is serially transferred to memory registers and operated upon using the shrink algorithm under the command of controls on the computer console. The computer applies the shrink operation to the image for a pre-set number of passes variable geometrically from 4 to 128. After this number of passes has occurred, the computer automatically reverts to a mode of operation wherein the tape continues to be read and the image is rewritten unchanged. At this time the computer may be ordered to count the number of isolated "1's" in the residual image. By alternating between the command to shrink and the command to count isolated "1's", the operator is able to complete the image histogram for the field of view which has been stored.

Another requirement of the computer is to complement the image stored. This routine is required in order to eliminate noise in the original image. Consider for example, the problem of image inclusions as in applying the shrink operation to the letter "O". Due to f(TUP) which indicates an X bit which is a link between two neighbors, this configuration of binary "1's" cannot be reduced to a single binary "1" by the shrink operation. This indicates that any inclusions which occur in a group of contiguous binary "1's" will prevent this group from being reduced to an isolated "1". In order to eliminate inclusions in the image received from the scanner the CELLSCAN computer first complements this image. In the complemented image inclusions appear as isolated "1's" or small groups of contiguous of "1's". These are eliminated by applying a modified shrink operation wherein f(ISO) = 0 so that isolated "1's" are not retained. Now the image is again complemented in order to recreate the original image in an inclusion free form.

Part of the computer is a test pattern generator which may take the place of the scanner as a data source. It is used during periods allocated to computer maintenance and debugging. The action of the computer is monitored by a video monitor which displays the original or residual image contained in computer memory. Furthermore, the action of the scanner is observed by a second monitor which is capable of displaying either the analogue video signal or the quantized signal.

Results

An example of an analogue signal from the scanner is shown in Figure 8, which shows a typical bilobe. Figure 8 shows the analogue signal before pulse stretching. It was obtained using a glass mounted blood sample prepared by the University of Rochester using peroxidase stain counterstained with Wright stain. An oil immersed 90 power apochromatic objective lens having a numerical aperture of 1.32 was used with a 1.40 numerical aperture oil immersed condenser. The virtual image formed by the objective was imaged on the photoconductive surface of a General Electrodynamic Corp. 7325 vidicon tube by means of a Leitz widefield periplant eyepiece. The light source was a tungsten spiral filament filtered by a 5300 Angstrom narrow band filter.

The analogue signal was quantized with the result shown in Figure 9. Inclusions in the nuclei due to non-uniformities in transmissivity can be seen. After elimination of inclusions by complementing the image and processing in the modified shrink mode this image would shrink, when recomplemented, to two isolated ones after about 50 to 100 passes.

Other cell types have been imaged and quantized using the CELLSCAN system. At this writing statistical information is being gathered on cells from many blood samples in order to ascertain what error rates can be expected in scanning bilobes in the presence of all other cell types.

General Considerations

In the above discussion of automatic blood cell scanning, we have described a machine which instruments a particular data processing routine. This special purpose machine is being used to ascertain whether the particular routine suggested is sufficiently general to identify binucleate lymphocytes. As in all discussions of special purpose pattern identifiers, the question arises as to how one is to determine the optimum approach to the problem at hand. Few treatments of the general theory of pattern recognition exist. Some that are known to the author are listed in the bibliography.[4,5,6] One of the more general treatises in this field is the one written by A. Gill.[6] He treats the general problem of pattern recognition by assuming a pattern set S, consisting of M subsets or individual patterns. He assumes that each of the M sub-sets contains N features. If we confine ourselves to a binary system, then the minimum value of N is $(\log_2 M)$, where the parentheses indicate the largest integral value of $\log_2 M$. Furthermore, the maximum value of N is equal to (M-1). In the latter case the M x N matrix characterizing the S is the unitary matrix, i.e., a matrix whose diagonal values are all binary "1's" and whose other elements are "0".

The more efficient the method of defining each M_i, the lower will be the value of N. Gill defines the efficiency of a noiseless pattern scanning system as equal to the information content of S divided by the value of N. Defining "e" as efficiency we thus have:

$$-\frac{\sum_i p_i \log_2 p_i}{M-1} \leq e \leq -\frac{\sum_i p_i \log_2 p_i}{(\log_2 M)}$$

where p_i is probability of finding the M_i in S.

Let us consider S as the population of white blood cells in peripheral human blood. The probability of occurrence of mononucleate lymphocytes, binucleate lymphocytes, and granulocytes respectively, are shown below:

$$p_{ML} = 3 \times 10^{-1}$$

$$p_{BL} = 3 \times 10^{-5}$$

$$p_G = 7 \times 10^{-1}$$

From the above figures, we can compute the information content of the source as equal to 0.70. Since we are dealing here with three patterns, Gill's approach would indicate both a maximum efficiency equal to 0.35. However, in the special purpose system which has been designed it should be noted that a value of N equal to 19,200 has been used. This indicates a system efficiency of 3.7×10^{-5}. This quantity is completely outside the range of efficiencies delineated by Gill. This would seem to indicate that a highly inefficient approach has been adopted in the present pattern analyzing system.

Contemplating this problem further, we should again note that, using the Gill approach, we should be able to obtain a value of N equal to 2. This implies that the existence or presence of only two features need be recognized by the scanner. For example, these two features could be:

A = The existence of two nuclei.

B = The existence of many granules.

The ideal scanner would recognize the bilobed lymphocyte as being characterized by $A\bar{B}$. All other lymphocytes would be characterized by $\bar{A}\bar{B}$. Granulocytes would be characterized by $(AB + \bar{A}B)$ or, merely, B. The problem now is how to design a scanner whose output can be directly translated into the existence or non-existence of the two features mentioned above. Gill's theory that such a scanner should exist does not give the designer a clue as to how such a scanner can be constructed. It appears clear to the author that existing contributions to the theory of pattern recognition require further extension in order to define methods whereby the salient features of a multiplicity of patterns may be determined. An alternative is to term the special purpose computer which accompanies the scanner as part of the scanner, rather than part of the recognition logic. This semantic manipulation, however, is of little value in solving the practical engineering problem.

Future Development

Some of the practical problems which must be solved in building a practical blood cell pattern analyzer may now be listed. As has been mentioned, the present system is a research tool for use in proving the value of certain data gathering and processing concepts. It operates at low speed and would, in fact, require over a month of continuous operation to process a single blood sample. A useful system is one which can process a blood sample (containing approximately 10,000 white cells) in about 15 minutes time. This rate of data processing allows about 100 milliseconds to process each cell. Taking into account present limitations of video scanning systems, about 30% of this time should be allocated to scanning and storing the cell pattern. This leaves approximately 70 milli-seconds during which to perform the shrink operation on the pattern stored for a sufficient number of passes to prepare an image histogram.

Let us assume that we extend the present serial mode machine directly and further assume that about 100 passes would be required for each cell identification. Noting that about 20,000 bits must be stored in each image, this implies that 2,000,000 bit operations must be performed in 70 milli-seconds, i.e., an allowance of 35 nano seconds per bit. In order to do logical operations at this rate we must work at the frontier of the state-of-the-art, using, let us say, a microwave memory as a data storage medium and high speed logic circuitry to process the data. Such a technique would require that the machine retrieve one bit of data from the delay line store, cause the operation of a logical sequence incorporating about 10 propagation times, and read out the results of this operation all in a 35 nano second period. Such a feat may well be beyond present day capabilities.[7]

Another approach would be to substitute a combined parallel-serial mode machine for the serial mode machine described above. For example, one might contemplate storing the image in a 300 word sequential access memory, having 64 bits per word. Parallel logic would be provided which would examine 64 bits of the image simultaneously. This would require a 64 fold increase in the number of logical gates which would be used to instrument the shrink logic. Some reduction in this figure is possible by cross connecting the f(TUP) logics. The computer would operate by storing three 64 bit words in its memory register and simultaneously storing 64 previously processed bits in an auxiliary register. A further 64 bit register would be required to store the output of the 64 shrink logic circuits. Because of the characteristics of the shrink algorithm, which at the moment is implemented by about 50 logic gates, it can be shown that a 4 to 5 fold increase in total machine complexity is indicated. The memory would perform a read-write cycle for every 64 bits of image processed. Again assuming 100 passes per image, this would imply 30,000 read-write cycles in 70 milli-seconds which is about 2 micro-seconds per read-write cycle. These specifications for a parallel-serial mode machine seem more reasonable than those corresponding to the serial mode machine. They do, however, imply a large increase in components required due to the adoption of parallel logic techniques.

Conclusion

In automatizing the analysis of blood cell patterns the computer engineer must devise machine techniques which can reproduce the human visual recognition process. It appears characteristic of such machines that vast amounts of input data must be operated upon in order to deliver a fairly elementary output. This is in contrast to present scientific and business computers where input and output data rates are commensurate. For example, in the present CELLSCAN system an input of approximately one quarter of a billion bits would be gathered in scanning a blood sample. This input is reduced to the quantity of bilobes in the sample which can be represented by a five-bit word. Even if the present system efficiency is improved to the maximum predicted by Gill, the ratio of input to output data rates would be of the order of ten-thousand to one.[6] Therefore, it is found that today's applications of data processing technology in the field of pattern recognition require us to take full advantage of present circuit art. It is here that use can be made of microwave logic techniques. Furthermore, a better theoretical grasp of pattern recognition problems is required so as to guide the engineer towards the most efficient use of available logic circuitry and data stores.

Acknowledgements

The author would like to acknowledge the support provided the CELLSCAN project by Dr. Marylou Ingram, of the University of Rochester, who has been the prime mover behind the medical investigation of the incidence and significance of binucleate lymphocytes. The project itself has been undertaken on a U. S. Atomic Energy Commission contract.

References

1. Ingram, Dr. Marylou, "The Occurrence and Significance of Binucleate Lymphocytes in Peripheral Blood After Small Radiation Exposures", International Journal of Radiation Biology, Special Supplement, Immediate and Low Level Effects of Ionizing Radiations (1959).

2. United States patent application number 854254, filed Oct. 8, 1959 by the Perkin-Elmer Corp.

3. Taylor, W. K., "An Automatic System for Obtaining Particle Size Distributions with the Aid of the Flying Spot Microscope", Brit. J. Appl. Phys., Supp. No. 3, 173 (1954).

4. Kirsch, R. A., et al, "Experiments in Processing Pictorial Information with a Digital Computer", Proc.EJCC, 221 (1957).

5. Unger, S. H., "A Computer Oriented Toward Spacial Problems", Proc. IRE 46, 1744 (1958).

6. Gill, A., "Theoretical Aspects of Minimal Scan Pattern Recognition", Electronics Research Lab., University of Calif., Series 60, Issue 233, March 23, 1959.

7. Turnbull, J. R., "100-Mc Nonsynchronous Computer Circuitry, Technical Digest, 1961 Internat'l Solid State Circuits Conf., Feb. 1961.

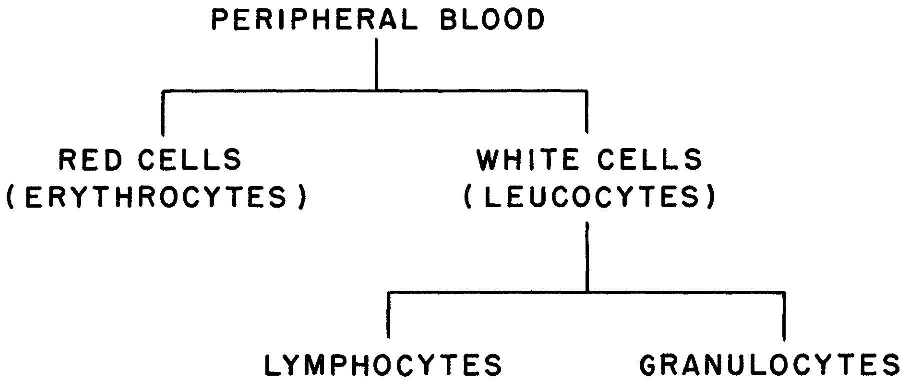


Figure 1. COMPOSITION OF PERIPHERAL BLOOD

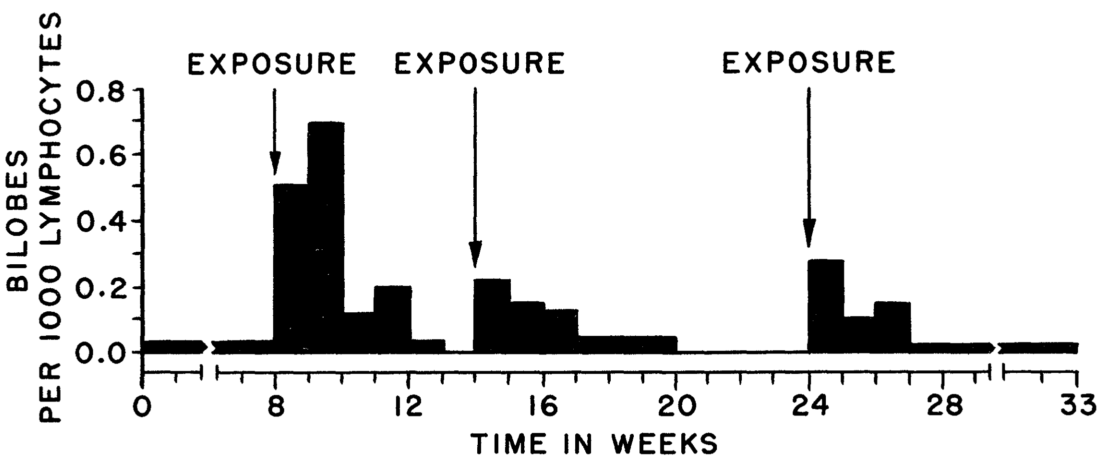


Figure 2. INCIDENCE OF BILOBES IN DOGS

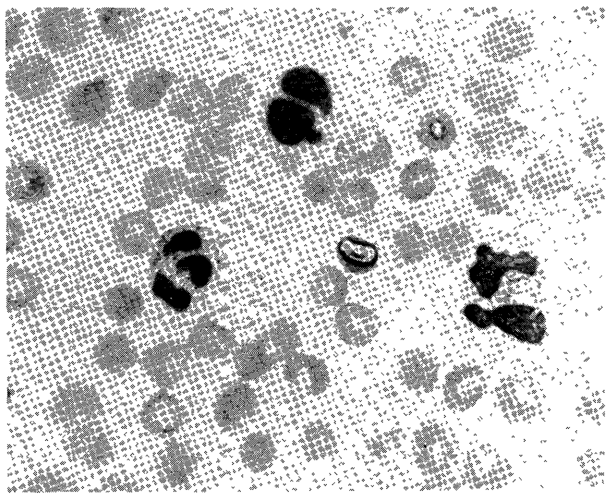


Figure 3. PERIPHERAL BLOOD SMEAR

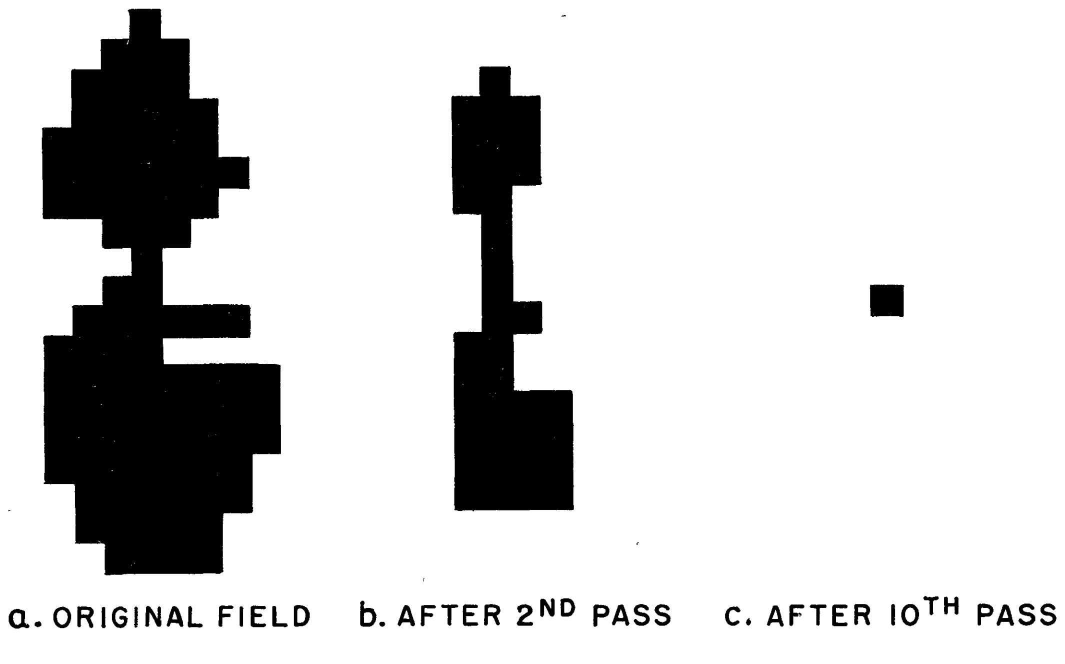


Figure 4. THE "SHRINK" PROCESS

A	B	C
H	X	D
G	F	E

$$f(\mathrm{ISO}) \equiv \bar{A}_P \bar{B}_P \bar{C}_P \bar{D} \bar{E} \bar{F} \bar{G} \bar{H}_P .$$

$$f(\mathrm{TAZ}) \equiv \bar{A}\bar{B}\bar{C} + \bar{B}\bar{C}\bar{D} + \bar{C}\bar{D}\bar{E} + \bar{D}\bar{E}\bar{F} + \bar{E}\bar{F}\bar{G} + \bar{F}\bar{G}\bar{H} + \bar{G}\bar{H}\bar{A} + \bar{H}\bar{A}\bar{B} .$$

$$f(\mathrm{TUP}) \equiv (A\bar{B} + \bar{A}B)\left\{(B\bar{C} + \bar{B}C)\left[(C\bar{D} + \bar{C}D) + \ldots + (G\bar{H} + \bar{G}H)\right]\right\}$$
$$+ \ldots + (E\bar{F} + \bar{E}F)(F\bar{G} + \bar{F}G)(G\bar{H} + \bar{G}H).$$

$$\text{FINALLY:} \quad f(X_P) = X\left[f(\mathrm{ISO}) + f(\mathrm{TUP}) + \overline{f(\mathrm{TAZ})}\right]$$

Figure 5. THE "SHRINK" ALGORITHM

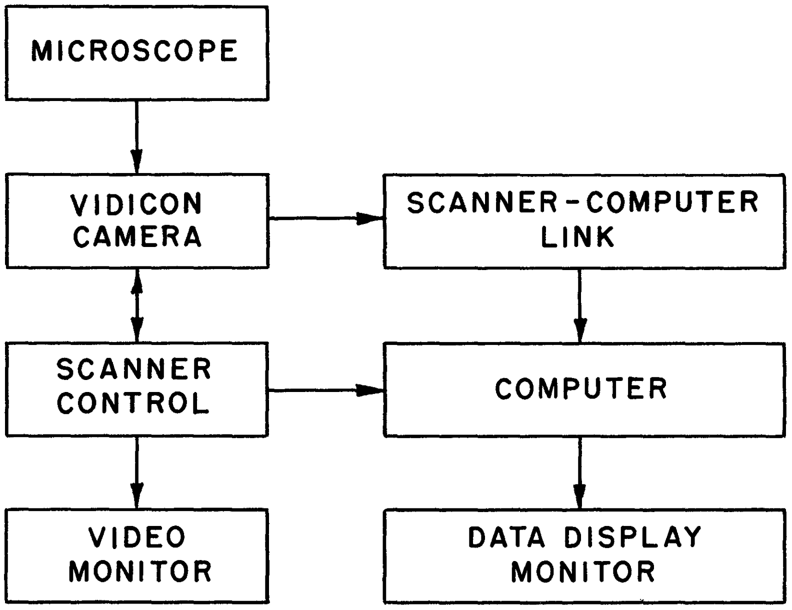


Figure 6. THE CELLSCAN SYSTEM

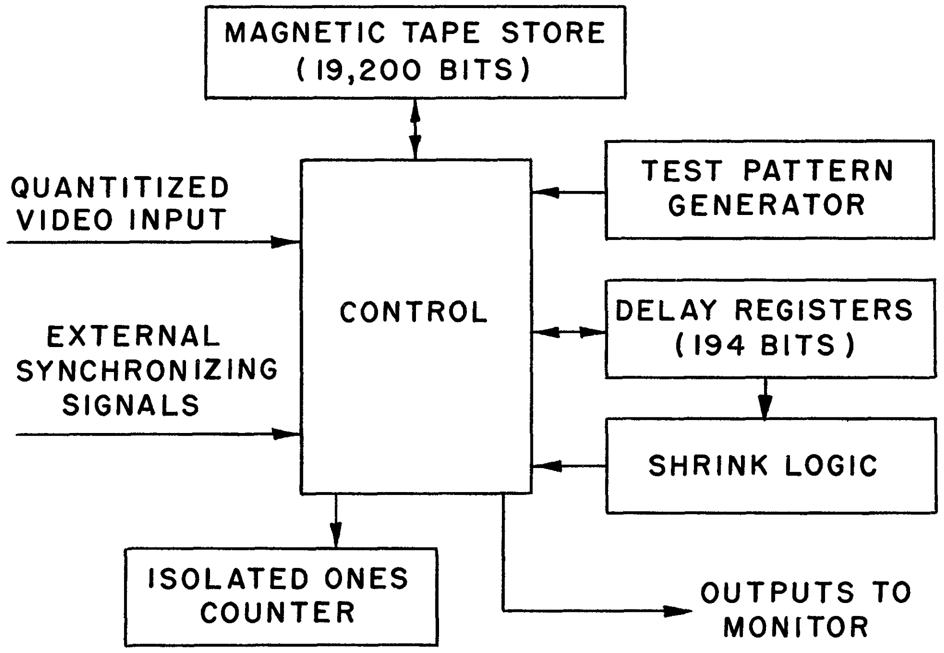


Figure 7. COMPUTER BLOCK DIAGRAM

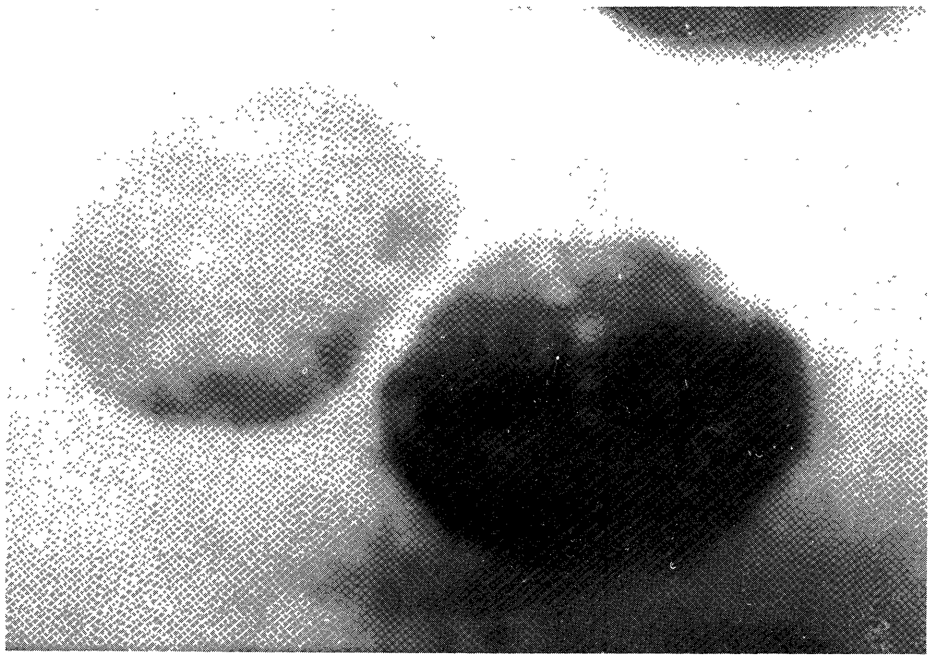


Figure 8. ANALOG VIDEO IMAGE OF BILOBE

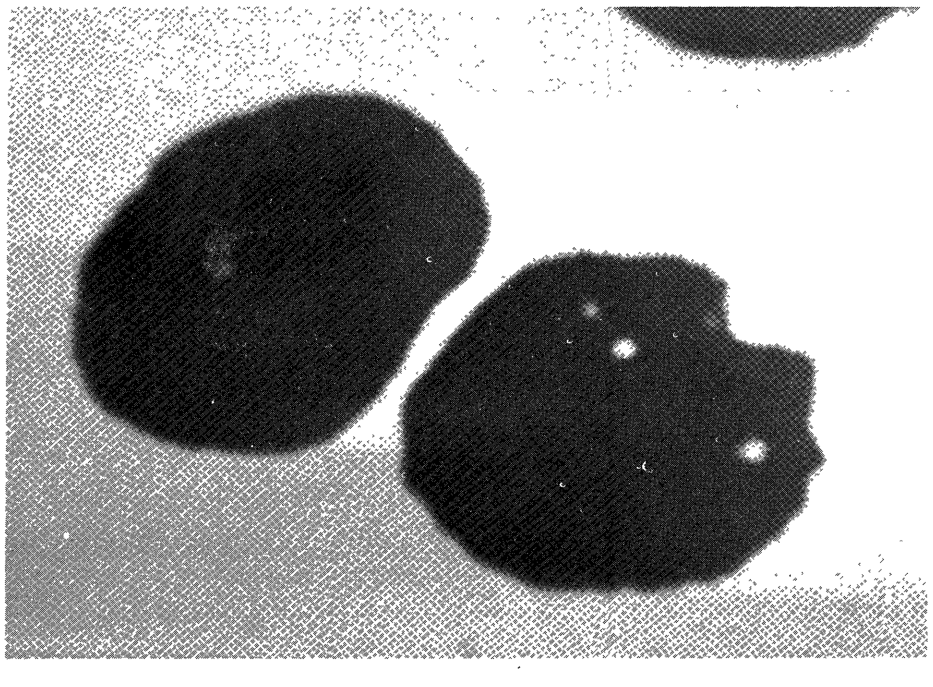


Figure 9. QUANTIZED IMAGE OF BILOBE

APPLICATION OF COMPUTERS TO CIRCUIT DESIGN FOR UNIVAC LARC

Gilbert Kaskey
Associate Division Director, Systems Design and Application
Remington Rand Univac
Philadelphia, Pennsylvania

Noah S. Prywes
Consultant to Remington Rand Univac
Assistant Professor, Moore School, University of Pennsylvania
Philadelphia, Pennsylvania

Herman Lukoff
Chief Engineer, Remington Rand Univac
Philadelphia, Pennsylvania

The design of circuits for computers has become in recent years a complex undertaking. The problem is two-fold. On one hand optimization of cost and speed is the prime objective. On the other hand, complexity is increased through factors such as component characteristics and life expectancy, manufacturing techniques, and the suppression of noise in very large systems. The complexity makes the use of computers as an aid to design almost imperative; this was the case in the design of circuits for Univac Larc.

Several applications of computers in circuit design are reported here and demonstrated by case histories for Univac Larc. The paper consists of two parts. In Part I, a general description of the problems and solutions is given. References to available reports or publications are given where a more detailed description can be found. The problem areas can be divided into three categories: evaluation of components and life test; design of circuits; protection against noise. Part II consists of a detailed description of the statistical techniques used in the circuit design.

Part I

A. Evaluation of Components and Life Tests

Evaluation of Components. The components evaluated included transistors, diodes, ferractors, ferrite cores, resistors, capacitors, etc. As an example, the evaluation of Philco surface barrier transistors will be reported here. This transistor was selected during the first half of 1956 after a study of many other candidates in respect to rise time, storage time, gain, current level at optimum performance and cost. The evaluation program required the testing of a large number of transistors to determine such parameters as beta, rise time, storage time, and breakdown voltage. These values were measured initially and at various times throughout life test.

In the attempt to mechanize the test data analysis, a complete library of Univac statistical routines has been developed. These statistical routines analyzed and evaluated the available data. The statistical analysis of empirical data is greatly simplified if the variate under analysis is normally distributed. Since this is rarely the case in practice, the distributions were transformed to ones that have Gaussian properties.

Close cooperation with manufacturers was maintained to insure that the transistors received were the best that could be produced in the manufacturing process used. Experimental designs were made to aid in the determination of the effect of various changes in production techniques, e.g., resistivity and etching time, on the several transistor parameters. A UNIVAC® routine was then used to perform the analysis of variance necessary for the identification of the statistically significant variables. The results, a joint effort between the manufacturer and user, helped establish production control procedures which virtually insured that component lots would meet the required specification.[5]

Evaluation of Life Test. Because of the long life expectancy of transistors it was difficult to ascertain failure characteristics by life test in a reasonable time period. In other words, it was not possible to detect significant degradation of transistor parameters over thousands of hours of life test. Since it was nevertheless extremely important to be able to make a prediction as to the life expectancy of the transistor, an attempt was made to run accelerated life tests at elevated temperatures, under severe humidity conditions and under vibration, with the purpose of producing gradual deterioration in a reasonable time. It was hoped that the correlation of such results with deterioration under normal usage conditions would result in a reasonably accurate prediction of life expectancy. The tests under elevated temperature conditions were the only ones that proved useful in this respect.

Transistors were placed on test at 55^oc, 65^oc, 75^oc, 85^oc, and 100^oc. The transistors involved were first tested for homogeneity by a study of the distribution of breakdown voltage and β. These parameters also appeared to be the major cause of transistor failure in the circuits and therefore are the subject of the investigation. By studying the behavior of these homogeneous sets over time, we hoped to obtain as estimate of transistor behavior at 25^oc.

To illustrate the statistical methods used, the determination of transistor life using degradation of breakdown voltage as a criterion will be discussed. The circuit design indicated that the breakdown voltage degradation to 3 volts implied a transistor failure. Theoretical studies had suggested the dependence of the breakdown voltage on temperature and time as follows:

$$\sqrt{V_p} = A - Be^{-a/2T}\sqrt{t} \qquad (1)$$

where V_p is the breakdown voltage, T is the absolute temperature and t is the age of the transistor in hours.

A least-squares fit was then made to the data, using fixed values of T, to determine the "best" values for A, B, and a. The results of this analysis are given in Table I.

Table 1. Punch-Through Voltage Accelerated Life Test

Group No.	Temperature (^{o}c)	Least-Squares Equation (t in hours)
10	55	10.6 - 0.0002t
11	65	9.57 - 0.0004t
12	75	11.9 - 0.0028t
13	85	10.9 - 0.0058t

Because of the assumed exponential relationship, the least-squares straight line was obtained for the logarithm of the slope as a function of reciprocal temperature. The equation thus obtained was

$$m = -1.3 \times 10^{15}e^{-(14278/T)} \qquad (2)$$

where m is a slope of the linear least-square equation and T is the corresponding absolute temperature as shown in figure 1-1.

This equation was used to estimate the slope at 25^{o}c, and the value was found to be approximately 2×10^{-5} volts per hour. The results of a fit by eye made prior to the regression analysis made on the Univac System indicated an average life of 211,000 hours as indicated in figure 1-2.

In addition to the prediction of accelerated life tests, life tests under conditions similar to operating conditions were conducted. Improved predictions were only possible as more data became available.

B. Circuit Design

Mechanization of the various steps involved in circuit design for Univac Larc has served as the basis for research whose objective is the complete mechanization of design and fault diagnosis of transistor circuits. Research has been in progress since the initial design phase of LARC attempting to mechanize all steps previously performed manually. Our objective has been to completely automate the design steps required in going from proposed circuit schematic configurations to the development of an optimized circuit. The process will consist primarily of computer programs using a detailed mathematical model. There are two advantages to such a process:

1) More efficient circuit optimization in terms of the predetermined functionals, that is, cost, by utilizing the speed of computers.

2) The generation of component specifications which computer programs correlate with the capabilities of the designed circuit.

Circuit design and optimization start with given circuit schematic configurations and a performance requirement. In the case of computer circuits, the latter can be stated, for instance, in terms of fan in (number of logical inputs), logical operation, fan out (number of circuits to which the output is connected), and the delay per circuit. The objectives of the design are to single out one of several suggested circuit configurations and determine component parameters so that cost is minimized.

The process can be roughly divided into three steps:

1) Generation of d-c circuit equations.
2) D-c circuit design.
3) Optimization of the Circuit for Reduced Delay.

Generation of d-c circuit equations. A code was devised for transferring the circuit information in a schematic diagram into the computer. This code is completely reversible; that is, the original circuit diagram can be derived uniquely from the computer code.

The nodes of the circuit are assigned a number m_1m_2. Each branch is uniquely determined by its endpoints (that is, the branch with nodes m_1m_2 and n_1n_2 as endpoints is called branch $m_1m_2n_1n_2p_1p_2$). An additional set of numbers p_1p_2 is necessary to differentiate two or more branches which have common endpoint notes. After listing a number which identifies a branch, the components of the particular branch are listed. When this has been done for all branches, the circuit has been completely described. Each component is associated with a letter of the alphabet (for example, resister R, emitter E, battery B, and diode D). In the format used each letter is followed by a two-digit number to differentiate components which are of the same type.

The component closest to node m_1m_2 is listed immediately after the description of the branch $m_1m_2n_1n_2p_1p_2$, followed in order by the remaining components in the branch; thus, the component next to node n_1n_2 is the last in the series. As an example, the following is the format for the circuit shown in figure 1-3.

00	01	00	R01	000
00	04	00	D00	G00
00	04	01	R00	B00
01	02	00	S00	000
01	04	00	R02	B01
02	03	00	Q00	000
02	04	00	E00	000
03	04	00	R03	B02
03	04	00	G01	000

This method of recording the information for a given circuit configuration gives a unique representation so that each component and its location is specifically described.

The two basic methods available for the generation of the circuit equations (that is, the loop and the nodal-branch techniques) have been considered. The loop method has the advantage of yielding fewer equations, since not all of the possible variables are included. If the additional variables are eliminated from the nodal-branch equations, the reduced set is identical with the set derived by using the loop method.

The nodal-branch derivation has a single advantage which is extremely important from the standpoint of a computer solution: the equations are derived very systematically. Thus the process of mechanization, which will result in a set of redundant equations, is easily implemented. On the other hand, when using the loop equation method, there are frequently too many loops to allow extracting those which make up the system of redundant equations.

It was decided, therefore, to concentrate on the more systematic nodal-branch method, and then eliminate any irrelevant variables.

If there are n nodes in the circuit, n-1 independent nodal equations can be generated. (It can be shown that if the n-th equation is generated, the result can be derived from the other n-1 equations.) Referring to the circuit in figure 1-3, the n-1 equations generated are:

$$I_{010} + I_{040} + I_{041} = 0$$

$$-I_{010} + I_{120} + I_{140} = 0$$

$$-I_{120} + I_{230} + I_{240} = 0$$

$$-I_{230} + I_{340} + I_{341} = 0$$

The branch equations represent the total voltage drop across each branch. For the sample circuit in figure 1-3 the equations are as follows:

$$V_1 - V_0 = V_{R01}$$

$$V_4 - V_0 = V_{D00} + V_{G00}$$

$$V_4 - V_0 = V_{R00} + V_{B00}$$

$$V_2 - V_1 = V_{S00}$$

$$V_4 - V_1 = V_{R02} + V_{B01}$$

$$V_3 - V_2 = V_{Q0}$$

$$V_4 - V_2 = V_E$$

$$V_4 - V_3 = V_{R03} + V_{B02}$$

$$V_4 - V_3 = V_{G01}$$

These four nodal and nine branch equations, then, represent a complete set of redundant equations which fully describe the system.

The current-voltage relation of diodes and transistors is nonlinear. In order to simplify computation, these nonlinear curves have been approximated and replaced by linear segments in the regions of operation which are of interest. Thus, whenever the voltage-current relationship of a diode is considered, the following condition is employed:

$$V_D = D_0 + R_D I_D$$

where V_D and I_D are the voltages and current respectively through the diode. (See figure 1-4.) The constants D_0 and R_D are unknown quantities to be determined in the calculations. In effect, a variable (V_D), which changes with input conditions, has been replaced by two quantities (D_0,R_D) which remain constant through varied input conditions.

In a similar manner the following substitutions can be made for the transistor currents:

$$V_S = S_0 + R_S I_S$$

$$V_Q = K_I (I_Q - K_2 I_S)$$

where S_0, R_S, K_1 and K_2 are constants determining the two straight line approximations; I_S and I_Q are the currents through S (base) and Q (collector) respectively; and V_S and V_Q are the voltage drops across the base and collector. (See figure 1-4.)

As in the case of the diode, unknown quantities (V_S, V_Q), which change with input conditions, are replaced by quantities (K_1, K_2, S_0, R_S) which remain constant through varied input conditions.

D-c circuit design. In the case of UNIVAC LARC the optimization of cost consisted mainly of reducing the required Beta of transistors used. There are two criteria for calculating the required Beta: Worst case design, statistical design.

In the case of worst case design, the parameters (such as resistances, supply voltages, etc.) are multiplied by a factor which represents the maximum tolerances allowed so that the Beta of the transistor involved becomes minimum.

In statistical circuit design, Monte Carlo or analytical[6,7], methods are used to obtain a distribution of the required Beta of the transistors as a function of the distributions of the circuit parameters (resistances, supply voltages, etc.)

Optimization of the Circuit for Reduced Delay. Examination of the above circuit equations shows that the number of unknown variables exceed the number of equations. Therefore there is no unique solution. Generally delay decreases with increase in Beta, although the delay would depend on many other parameters as well. The purpose of the optimization is then to determine a unique circuit having the lowest Beta requirement such that the maximum delay allowed in the circuit specifications is not exceeded.

The transient behavior of the circuit can be determined experimentally, analytically, or through statistical studies.

In the experimental transient study the unknowns in the circuit equations are divided into so-called dependent variables and independent variables. The number of the dependent variables is equal to the number of equations. The determination of optimum values for the independent variables inplies unique solution of the circuit equation which represents the optimized circuit. The problem then is to vary the independent variables and determine experimentally the values corresponding to minimum delay. This can be an iterative process where one of the independent variables is varied while the others are kept constant.[8] Figure 1-5 illustrates such an experiment where R3 is varied for a transistor Beta of 9 and the on-base current equals 1.15 ma. The optimum value of R3 is found to be approximately 750 ohms.

A large number of circuits have to be computed in the process of optimization using circuit equations modified for worst case design. These circuit equations are found to be nonlinear. Solution of the system of equations by computer is of significant advantage over manual computation, especially when the system of equations is nonlinear.

The theoretical relationship between circuit delay and the several circuit parameters has been found to be extremely unreliable for prediction purposes and therefore abandoned.

The third approach involves the statistical determination of the relationship between the circuit parameters and the circuit delay. Specifically, the determination of the regression of circuit delay on the transistor parameters has been established. This method is the key to the circuit design procedure used and therefore it is described in considerable detail in Part II of this paper.

A functional relationship, developed statistically, serves two closely related purposes. It is not only necessary for any work in statistical circuit design but offers the following advantages:

1) Changes in production control, which, experience indicates, occur frequently, may improve the parameters of the selected transistors in some respects and degrade them in others. Also, with the rapid developments in transistor production, newer, better and less expensive transistors become available. A correlation between transistor parameters and circuit performances allows the transistor manufacturer the freedom of changing production controls to improve one parameter at the expense of others so that transistors improve in yield and cost. Also a freedom is maintained to purchase transistors from many sources.

2) The circuit that has been designed for a particular specification may be useful in other applications in which a less expensive transistor would satisfy a functional specification calling, for example, for slower speed or less gain. The regression of circuit performance on the several component parameters allows such a change to be made without additional design or experimental check.

C. Reduction of Noise and Delay in Backboard Wiring

The transmission delay increases with the lengths of the wire and distributed capacitances representing connectors, wires, etc. The noise pick up (that is, voltages and currents induced in a wire by pulses in other wires in its proximity) increases with the lengths of the wires but decreases with the total distributed capacitance on the wire. The assignment of elements on the backboard is made to reduce delay and noise pick up. In Univac Larc, the logical designer decided where groups of circuit elements were to be placed on the backboard, based on his familiarity with general information flow path among organs of the computer. The circuit elements were then assigned to specific printed circuit packages. Preliminary wiring procedures were then run on Univac I to determine whether bad cases, representing wires exceeding length or capacitance, existed. This was done on the basis of wire length calculation between terminals, and calculation of total capacitance represented by connectors and wire lengths. When bad cases existed, the logical elements were moved (by decision of the logical designer) in an attempt to reduce and/or eliminate the bad case conditions.[9] Iterations of this procedure, partly manual and partly automatic, are continued until wire lengths and capacitances are reduced to a tolerable level.

Research on computer placement of circuit elements on the backboard has continued after completion of the layout of Larc-Univac. A suitable algorithm has since been developed for performance of this task.[10] The algorithm is capable of minimizing the longest wires on the backboard or the total length of all wires combined.

Part II

The engineer who has as his assignment the design of a transistor circuit to perform according to a predetermined functional specification has a choice of two courses in designing the circuit and specifying the transistor.

One approach, in general use, is to determine by measurements the worst parameters of a selected type of transistor, to employ these parameters as the limiting criteria in a worst-case design, and to choose the other components in the circuit to optimize speed or gain, for example, in a nonrigorous way.

In the second approach, discussed in this paper, the engineer designs a circuit for a typical transistor which performs to the given functional specification. The other circuit components are selected to optimize the operation with this transistor. The dependence of the functional operation of the circuit (for instance, its gain or delay) upon the parameters of the transistor is determined over a wide range of variations of these parameters through statistical studies. Transistor parameters are then determined for each range corresponding to the functional specifications.

Information on the dependence or correlation of parameters is valuable to both the transistor manufacturer and the circuit designer. One transistor parameter can be improved at the expense of another so that the transistor improves in production yield, cost, etc., without harmful effect on the operation of the circuit. A change in production control, rather than being harmful, can be helpful. Various types of transistors are candidates for use in the circuit without additional design or experimental work, and the same circuit can be used to satisfy a number of specifications, changing only the type of transistor. The circuit designer can use the same information for statistical circuit design[4] as opposed to the worst case design, thus effecting additional savings.

The subject approach will be illustrated by a case history of a circuit design. To relate a given circuit specification to the transistor parameters involved, a considerable amount of computation is necessary, which has been carried out on a UNIVAC I data-processing system.

A. Description of the Circuit

The functional specifications of the circuit to be designed were as follows:

Driving circuit: Flip-flop whose output voltage has exponential rise time to 70% in 40 mμs.

Input voltage: Pulse from -2.9v to -0.3v

Input current: Pulse from 0 ma to 4.5 ma.

Output voltage: Pulse from -2.9 to 0v.

Output current: Pulse from 0 ma to 52.0 ma.

Maximum output capacitance: 1000 μμf.

Maximum load: 32 standard circuits.

Delay:	Minimum	Maximum
High-speed range	22 mμs	165 mμs
Medium-speed range	33 mμs	205 mμs
Low-speed range	44 mμs	245 mμs

The circuit configuration chosen is shown in figure 2-1. Delay is measured from the beginning of the clock pulse driving the flip-flop to the beginning of the output of the leading circuits. The delay-measuring circuit is shown in figure 2-2.

Since the minimum delay is not critical in this configuration, design effort continued with the input and loading for maximum delay, as shown in figure 2-3. Measurements were made when the transistor was turning off, since maximum delay occurs at that time. A typical transistor was selected to give a delay in the medium range. The values of the other components in the circuit that would minimize delay were then determined experimentally. The Surface Barrier Transistor (SBT) in the circuit (figure 2-1) has a relatively small effect on the delay of the entire circuit; therefore, the determination of worst parameters for the SBT was feasible. This paper will deal with the regression of the parameter of the second transistor and on the performance of the circuit as a whole.

Like the choice of the circuit configuration (figure 2-1), final determination of the values of components other than transistors was based upon other experimental work not relevant to the work discussed here.

B. Parameters of the Transistors

Four parameters and circuit delays were measured for each of 360 transistors. The six parameters normally specified by the manufacturer are Breakdown voltage, Leakage current, Current gain (β), Rise time (T), Storage time (S), and Peak base-to-emitter voltage (V). The first two, which affect mainly the dc operation of the circuit, have no significant effect on circuit delay. The remaining four parameters, assumed a priori to affect circuit delay significantly, were measured in each transistor. Current gain (β) was measured at a constant collector current of approximately 100 milliamperes and a collector voltage of -0.6. Because of dc considerations, gain must exceed 30 under these conditions. The circuits shown in figures 2-4, 2-5, and 2-6 were used to measure T, V, and S, respectively. Total circuit delay (δ) was assumed to be a function of V, β, T, and S. In addition, δ was measured for each transistor, with output loading which corresponds to maximum delay (figure 2-3).

The transistors tested were in three groups: 236 transistors of type GT762 (taken from two production runs), 99 transistors of type CK762, and 25 transistors of type TA1830.

No theoretical relationship among the measured parameters was assumed. Each measurement was performed twice. Transistors whose values did not check within the accuracy limits of the measuring device, were eliminated from further consideration but the number of these was negligible.

C. Statistical Studies

Interdependence studies between the parameters and delay values were undertaken first. The tools of regression analysis[1] were used to ascertain whether, in general, circuit delay can be predicted from known parameters of a transistor. The second step was the establishing of a functional relationship between circuit delay and the several known parameters. This function formed the basis for the successful determination of transistor parameters for the delay ranges.

Studies in Regression Analysis. To investigate whether there is any direct relationship between circuit delay (δ) and any of the four transistor parameters listed, the measured value of circuit delay for 236 transistors, assuming these to be a representative sample of the population of all GT762 transistors, was plotted against each of the parameters. It was assumed that the regression of δ on each of the parameters V, β, T, and S is linear. The scatter diagrams of δ versus each of the transistor parameters and fitting of linear regression equations are given in figures 2-7 thru 2-10. The mean δ values are connected in a line of best fit, shown as a light line; the line of regression, shown as a heavy line, is defined as the linear function of the form $y = mx + b$, which fits the means of arrays best, in the least squares sense. The fitted linear regression equations are given below:

$$\begin{aligned}\delta &= -33.4V + 194\\ \delta &= -0.0812\beta + 197\\ \delta &= 0.577T + 146.5\\ \delta &= 12.4S + 171\end{aligned}$$

By using t tests[2], it was found that the regression between δ and V is not statistically significant but the regressions of δ on the other parameters are highly significant, i.e., at the 1% level. There is a definite indirect relationship, then, among β, T, and S. The indirect relationships are obtained by using the last three of the above equations. Though the estimated coefficient of regression between δ and V is greater than the corresponding coefficient between δ and any other parameter, it is not statistically significant since the estimated variance of this coefficient is very high. Further investigation made to determine whether any direct relationship between the parameters V, β, T and S existed, indicated a strong direct relationship between both V and T and also between T and β.

Delay as a Function of Sixteen Expressions. A UNIVAC program was used to find the linear fit and regression coefficients between δ and the following 16 parameter expressions:

$V, \beta, T, S, 1/\beta, T/\beta, T^2/\beta, T/\beta^2, V/\beta,$

$VT, TS, S/\beta, T^2, 1/\beta^2, 1/S, VS$

It was assumed that the coefficients of regression between δ and other parameter expressions were not significant. The program revealed highest positive regression between δ and terms T, T^2, T^2/β, VT, S/β, and TS. These six parameter combinations were chosen for a function with linear constants as follows:

$$\delta = K_1T + K_2T^2 + K_3VT + K_4T^2/\beta + K_5S/\beta + K_6TS + K_7 \quad (1)$$

A second program was subsequently written to apply the least-squares fit criterion to the 234 sets of transistor data for the given equation. The normalized equations (seven equations, seven unknowns) of the fit were solved by the Crout Method[3]. A third program was developed to test the curve fit; that is, to compare the calculated δ with the observed and to determine the individual term contributions. These programs revealed that the K_5S/β and K_6 TS terms contributed little to the value and could be dropped, thus simplifying the function to the following:

$$\delta = K_1T + K_2T^2 + K_3VT + K_4T^2/\beta + K_7 \quad (2)$$

where $K_1 = 1.666$, $K_2 = 0.001$, $K_3 = -2.717$,

$K_4 = -0.175$, and $K_7 = 129.135$

A relatively simple evaluation of the normality of the distribution of errors based on this regression equation is indicated in figure 2-11. The cumulative distribution of errors would appear perfectly linear in the representation of a normally distributed population.

Discussion of Accuracy of Prediction Using the Function. A lot of 99 type GT762 transistors from a later shipment was measured to determine the applicability of the derived functions, equations (1) and (2). The distribution of errors between the equation prediction (2) and the observed values of delay appeared normal, with a mean of 8%. A change in the constant term or inclusion of dependence on S would correct the function as applied to this particular group and shift the mean to zero.

D. Range Determination

Since the circuit under design specifies use in one of three delay ranges, rather than a specific delay, a method for classifying transistors into the three ranges according to known parameters would serve the purpose. The method capitalizes on the relationship established in the search for a predictive function.

The 236 units first investigated were plotted on a T-ordinate, β-abscissa graph, and labeled with their observed δ values. Arbitrary δ ranges were found to separate themselves fairly well into various regions of the plot; rough borders were sketched between regions following the best range separations. These T (β) curves descended exponentially at low β values, and leveled off horizontally as β increased, suggesting the functional relationship:

$$T = K_1 + K_2 e^{-K_3 \beta} \qquad (4)$$

The δ-labeled points were separated into the three designated groups: 0-155 mμs, 156-195 mμs, and 196-234 mμs; and the two borders were added. A program was devised to fit the border β, T data to the suggested functional expression, yielding the constants K_1, K_2, K_3 for each curve. The smooth exponential decay curves were drawn in to separate the data. The results, for the first lot of 234 type GT762 transistors, are described as follows:

1) In the high range ($196 \leq \delta \leq 235$), eight units out of 52 occurred which did not belong. Their values were 180, 180, 186, 192, 192, 192, 192, 194 mμs. Thus there were only two outside of the tolerance criterion, ± 10 mμs. This tolerance was selected arbitrarily by adding the measurement tolerances of T and δ, each ± 5 mμs.

2) In the medium range ($156 \leq \delta \leq 195$) nine units out of 179 occurred which did not belong. One unit was below (152), and eight units were above (196, 196, 196, 198, 198, 198, 200, 200). None of these was outside the ± 10 mμs tolerance region.

3) The low range ($\delta \leq 155$) contained only two units, both of which were correctly placed.

4) The two border equations are as follows:

$$T = 30.97 + 60.58e^{-0.0157\beta} \text{ at } \delta = 155 \qquad (5)$$

$$T = 62.18 + 112.41e^{-0.0166\beta} \text{ at } \delta = 195 \qquad (6)$$

The results suggest a very accurate separation. The lowest region, where there was insufficient data available, was checked on another set of transistors. The results are discussed below.

Figure 2-12 is a graph that can be used to sort transistors by β and T measurements. Once the measurements for each transistor are made, the β-T point on the graph establishes the delay range of the unit.

Discussion of Accuracy of Prediction Using Ranges Determined. With the method just indicated, using the transistor delay-range chart with the originally derived borders (figure 2-12), the new lot of 99 type GT762 transistors was plotted. There were 28 transistors in the high range ($196 \leq \delta \leq 235$).

In the medium range ($156 \leq \delta \leq 195$), 69 units occurred, of which 11 did not belong. Nevertheless, of these 11, ten units were acceptable under the tolerance limits, indicating only one misplaced.

In the low delay range ($\delta \leq 155$), only two transistors occurred, both of which were correctly placed.

A linear shift in the borders of the δ ranges would take care of the errors of misplacement. These results are strongly indicative that new lots of the transistor have some property changes that can affect our application, unless additional parameters such as storage time (S) are considered.

An excellent prediction for the TA1830 data was achieved by the transistor delay-range chart (figure 2-12). Of the 25 units tested, 22 fell within the predicted range and 3 were borderline. The borderline cases were so close that, within tolerance limits, they could be placed in the correct categories.

In contrast to the broad range of delay values in the original 236 type GT762 transistors, these RCA TA1830 units were mostly confined to the lowest delay range.

Acknowledgements

The authors gratefully acknowledge the assistance of the many groups and departments of Remington Rand Univac who contributed to the success of this paper. Special acknowledgement is given to P. Krishnaiah and P. Steinberg.

References

[1]Ezekiel, Mordecai. Methods of Correlation Analysis. 2nd ed. New York: Wiley, 1941.

[2]Johnson, P. O. Statistical Methods in Research Chap. V. Prentice Hall, Inc., 1944.

[3]Hildebrand, F. B. Introduction to Numerical Analysis. p. 429. McGraw Hill, 1956.

[4]Gray, Harry J., Jr. An Application of Piecewise Approximations to Reliability and Statistical Design and Proceedings of IRE. July, 1959.

[5]Remington Rand Univac, Division of Sperry Rand Corporation. Statistical Techniques in Transistor Evaluation Final Report. Dept. of the Navy, Bureau of Ships NObs 72382. Applied Math Department: Remington Rand Univac, Philadelphia, Pa. April, 1959.

[6]Benner, A. H., and Meredith, B. Designing Reliability into Electronic Circuits. Proc. Nat'l Electronics Conf. Vol. 10, pp 137-145. Oct. 1954.

[7]Gray, H. J., Jr. An Application of Piecewise Approximations to Reliability and Statistical Design. Proc. of the IRE Vol. 47, No. 7, pp. 1226-1231. July, 1957.

[8]Remington Rand Univac, Division of Sperry Rand Corporation. Univac Larc Highspeed Circuitry Case History in Circuit Optimization. Prywes, N. S., Lukoff, N., and Schwartz, J. Remington Rand Univac, Philadelphia, Pa.

[9]Remington Rand Univac, Division of Sperry Rand Corporation. The Univac Prepared Engineering Document Program. Williams, T. Remington Rand Univac, Philadelphia, Pa.

[10]Remington Rand Univac, Division of Sperry Rand Corporation. The Backboard Wiring Problem: A Placement Algorithm. Steinberg, L. Remington Rand Univac, Philadelphia, Pa.

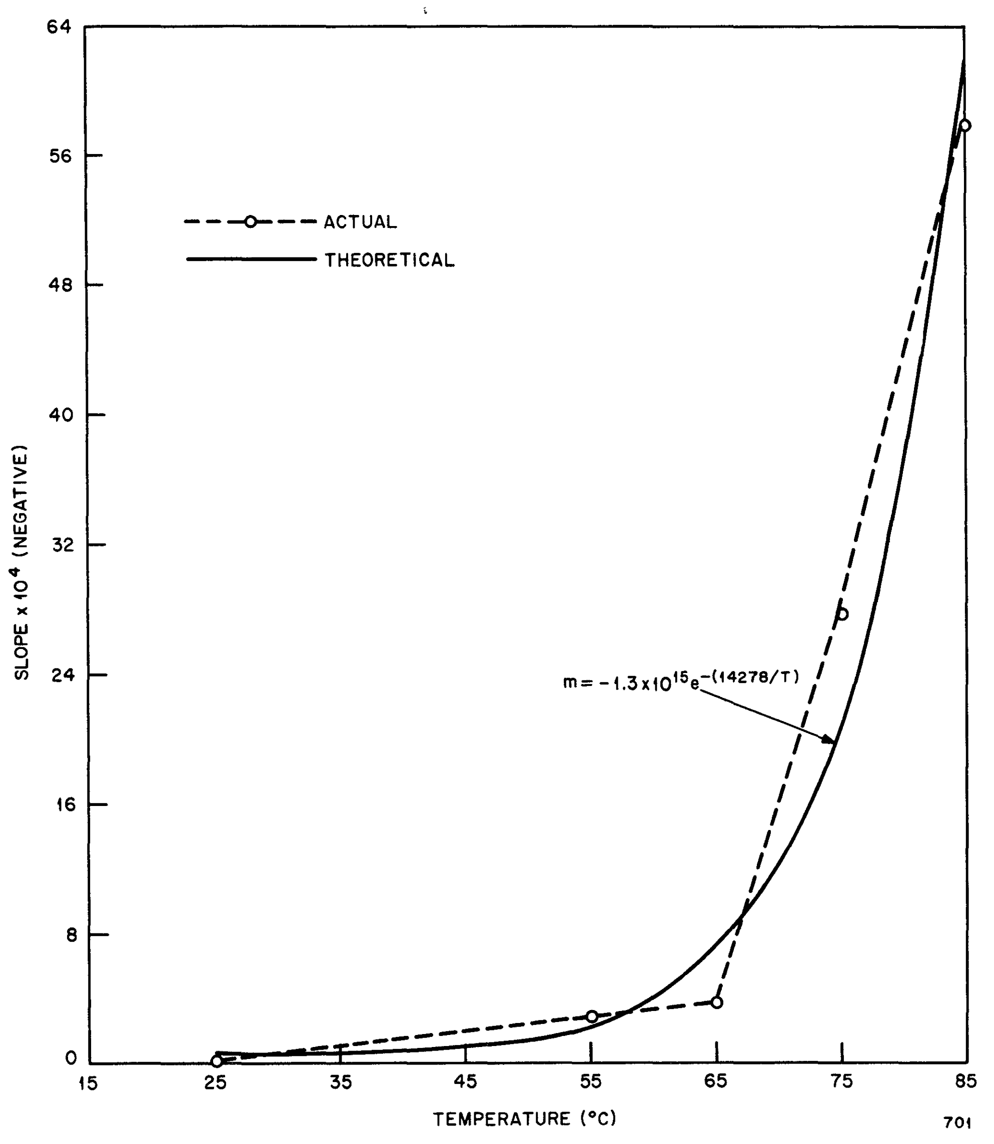


Figure 1-1. PUNCH-THROUGH VOLTAGE SLOPE PREDICTION

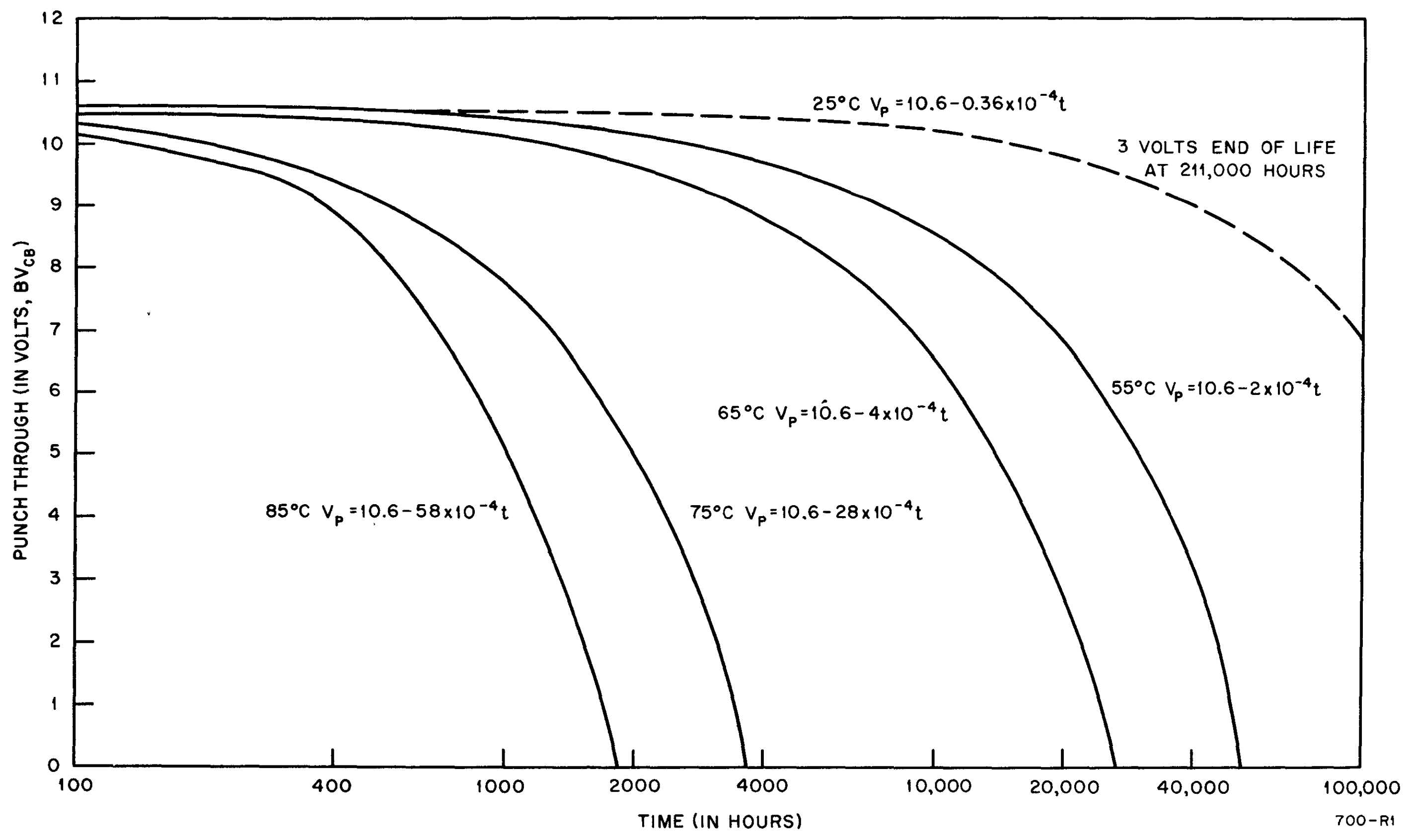


Figure 1-2. PUNCH-THROUGH VOLTAGE ACCELERATED LIFE TEST

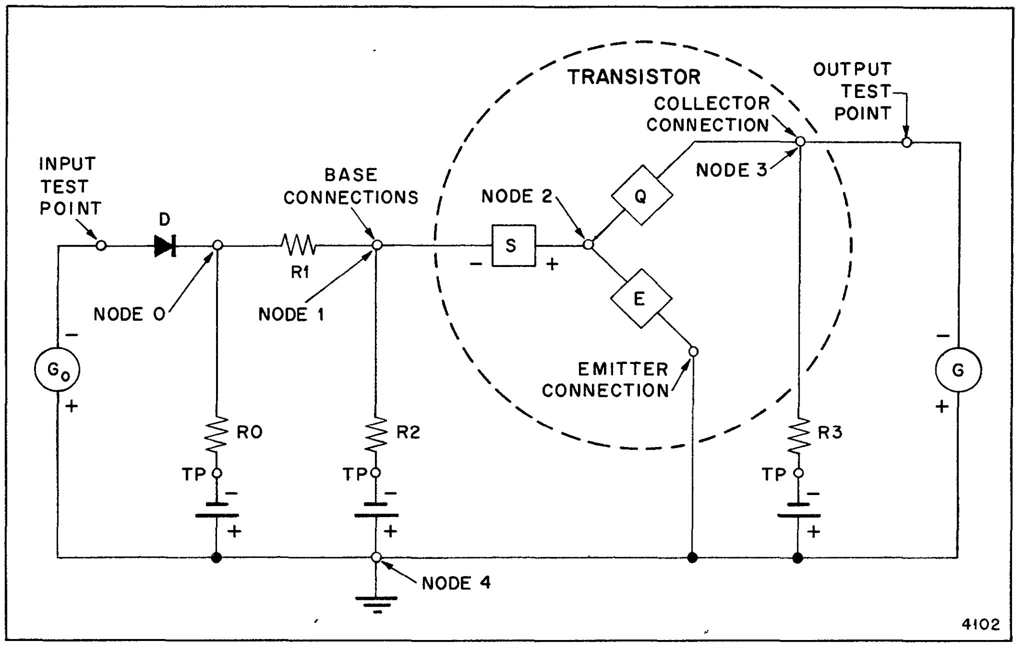


Figure 1-3. SAMPLE CIRCUIT FOR FAULT DIAGNOSIS

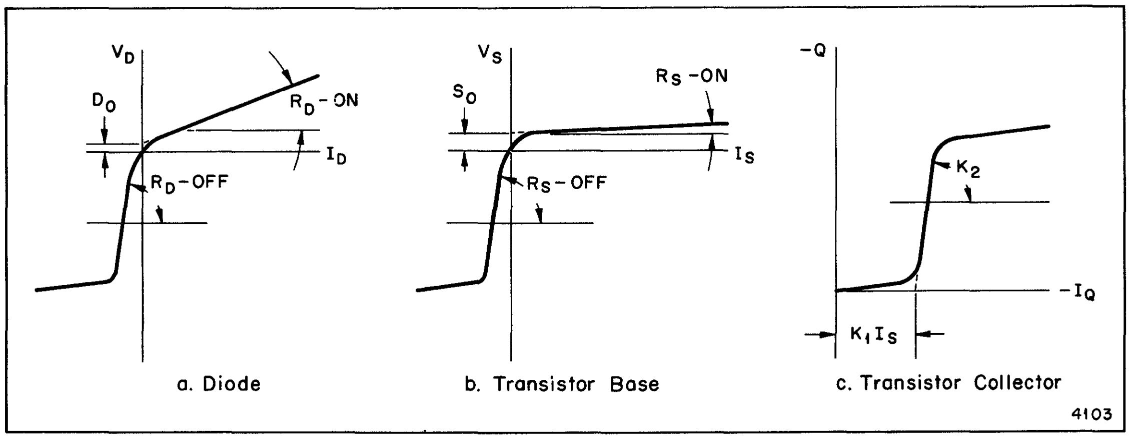


Figure 1-4. VOLTAGE-CURRENT CHARACTERISTICS OF A DIODE, AND BASE AND COLLECTOR OF A TRANSISTOR

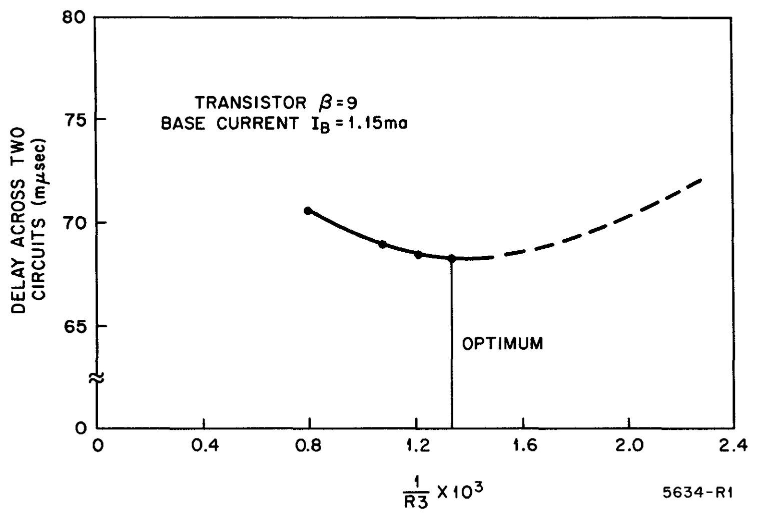


Figure 1-5. DELAY AS A FUNCTION OF THE 1/R3

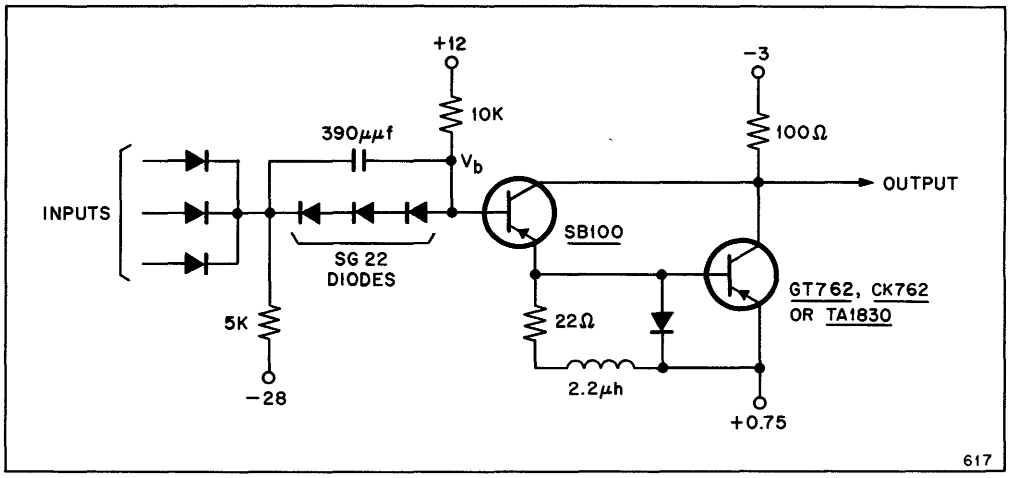


Figure 2-1. SCHEMATIC DIAGRAM OF CIRCUIT

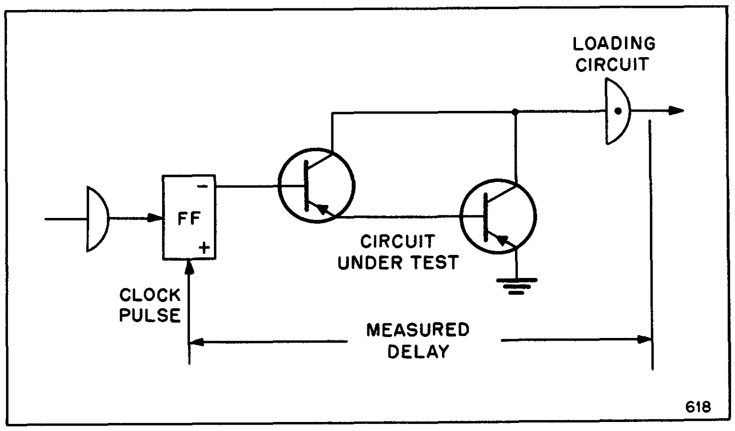


Figure 2-2. DELAY MEASUREMENT

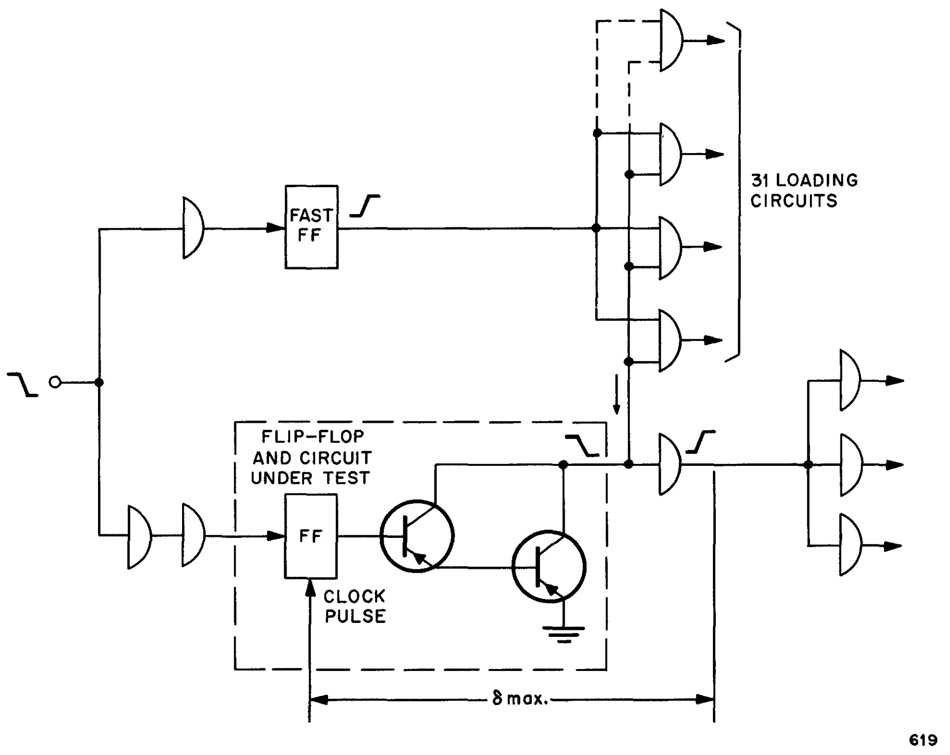


Figure 2-3. INPUT AND LOADING FOR MAXIMUM DELAY

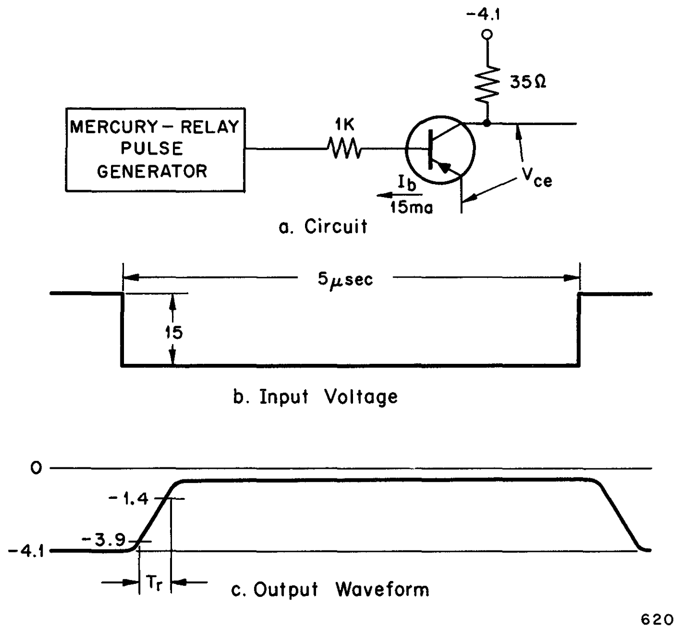


Figure 2-4. MEASUREMENT OF RISE TIME

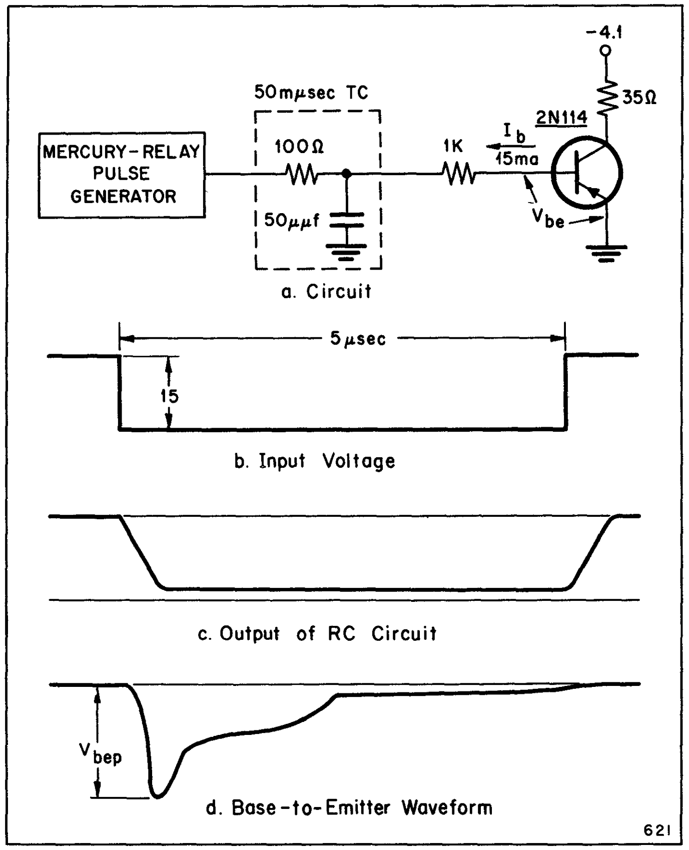


Figure 2-5. MEASUREMENT OF PEAK BASE VOLTAGE

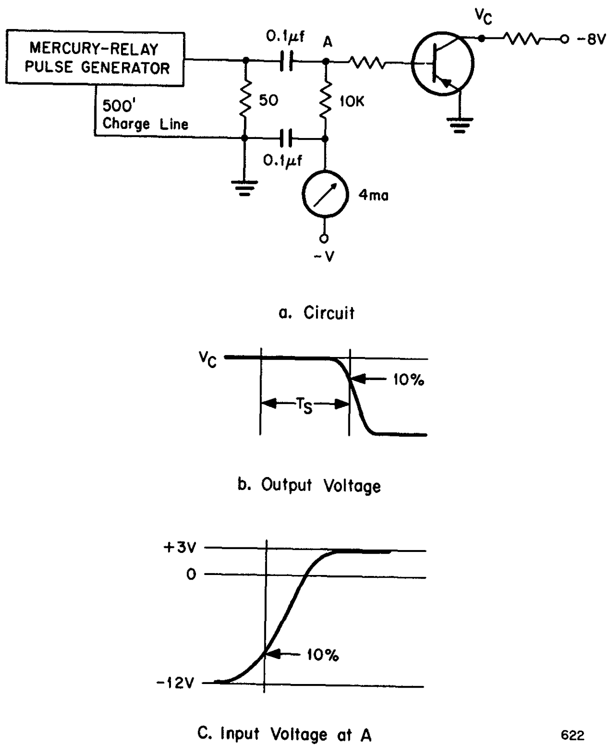


Figure 2-6. MEASUREMENT OF STORAGE TIME

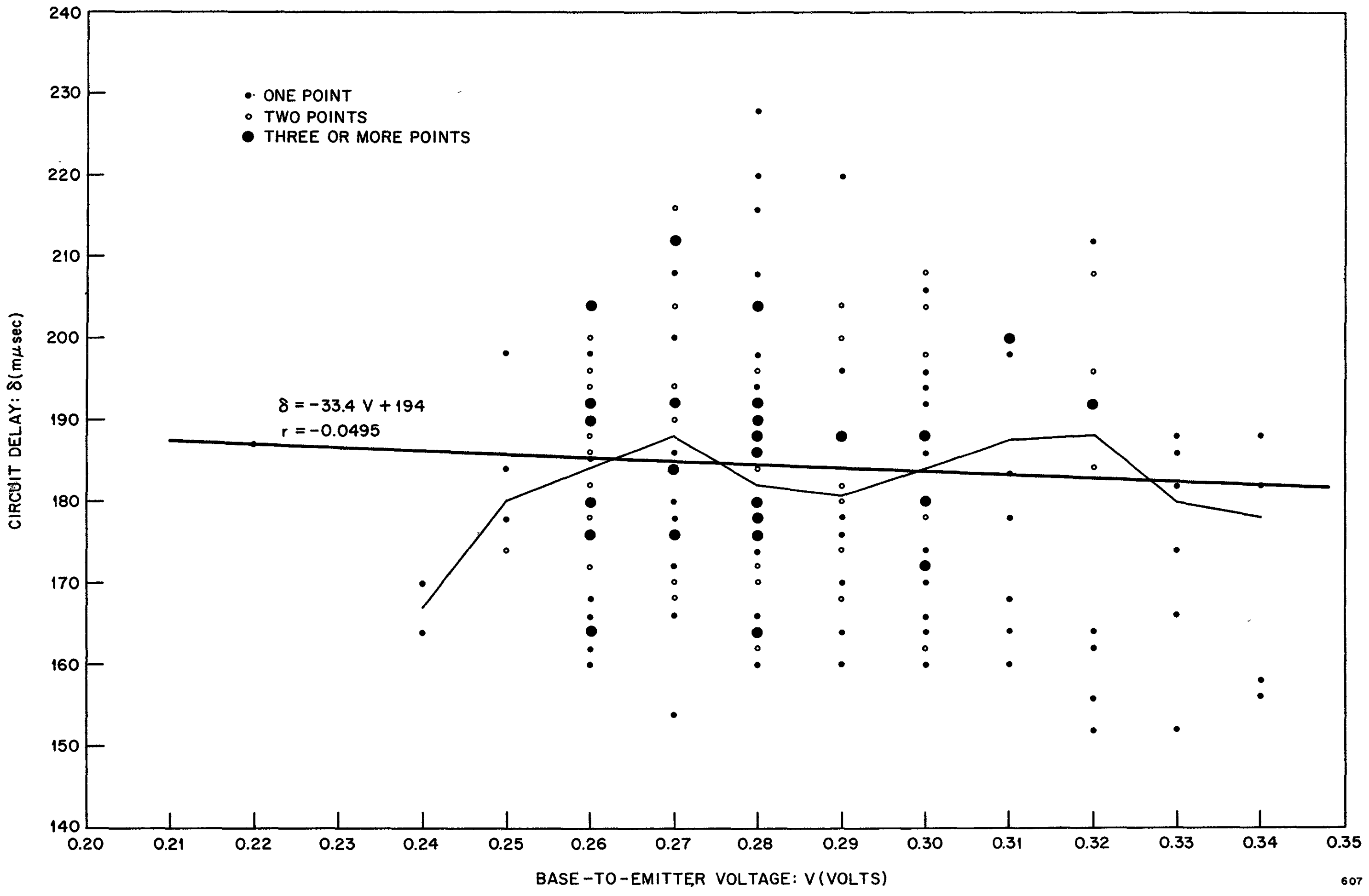


Figure 2-7. SCATTER DIAGRAM AND REGRESSION LINE FOR CIRCUIT DELAY AS A FUNCTION OF BASE-TO-EMITTER VOLTAGE

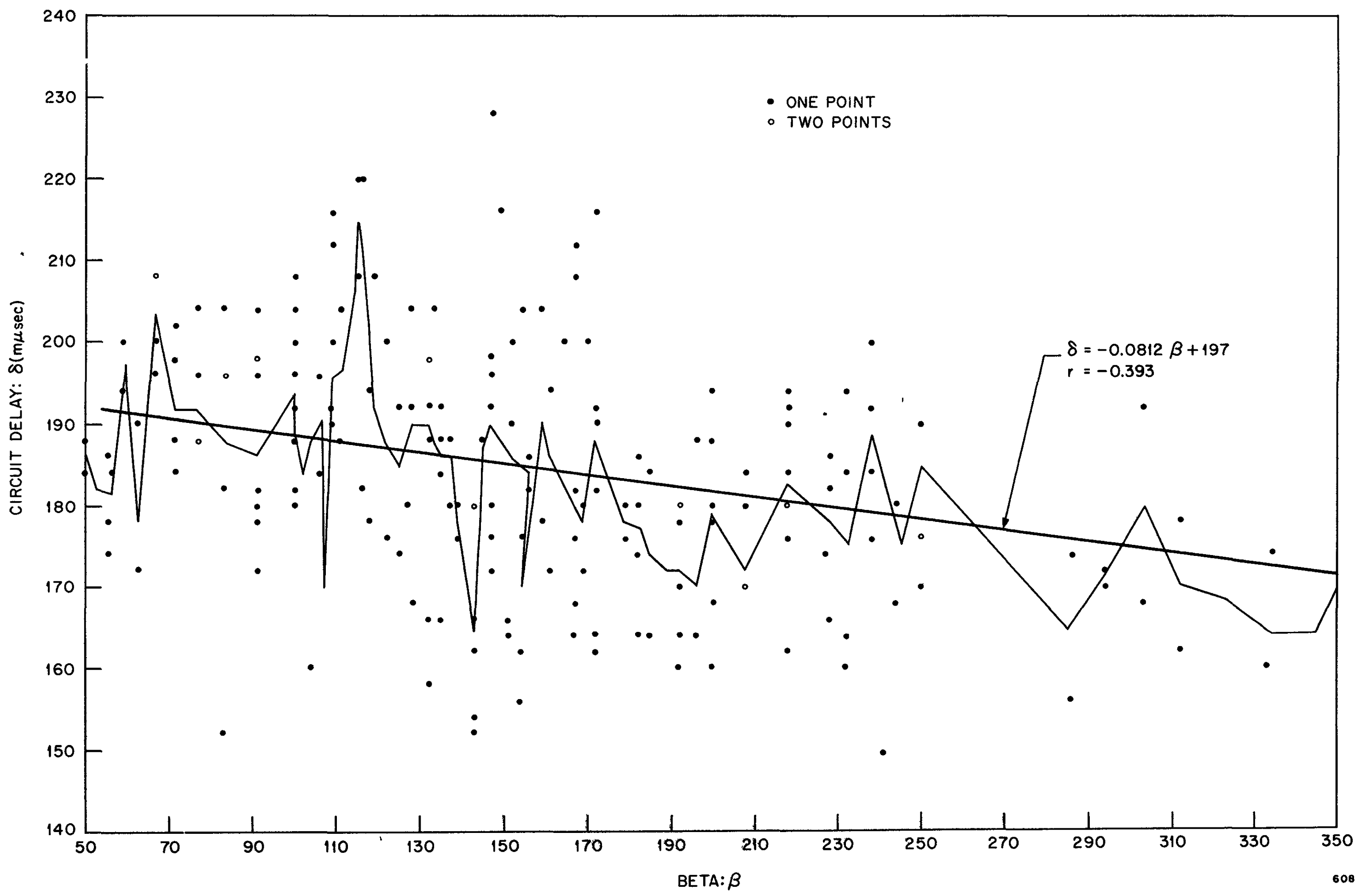


Figure 2-8. SCATTER DIAGRAM AND REGRESSION LINE FOR CIRCUIT DELAY AS A FUNCTION OF BETA

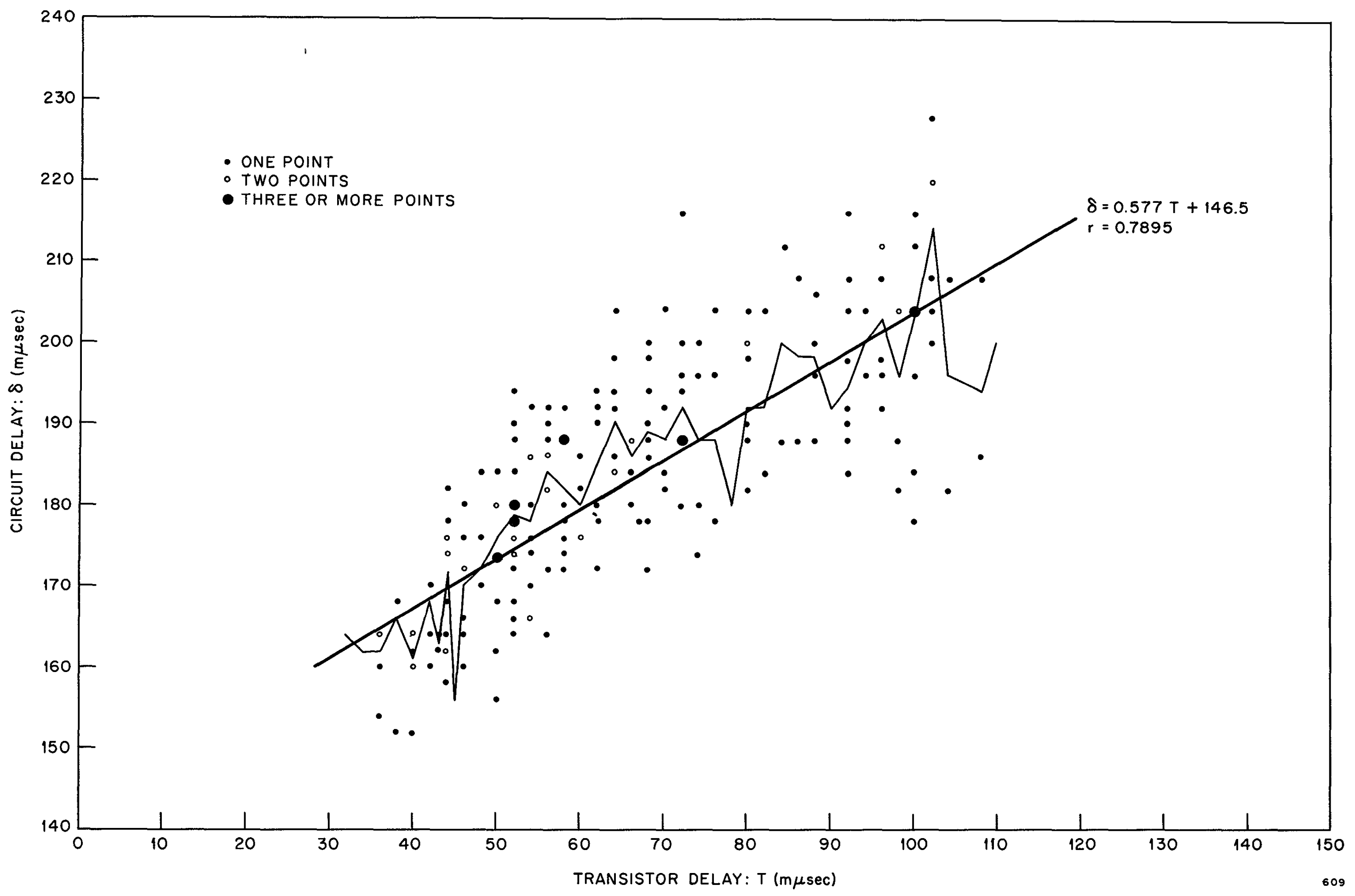


Figure 2-9. SCATTER DIAGRAM AND REGRESSION LINE FOR CIRCUIT DELAY AS A FUNCTION OF TRANSISTOR DELAY

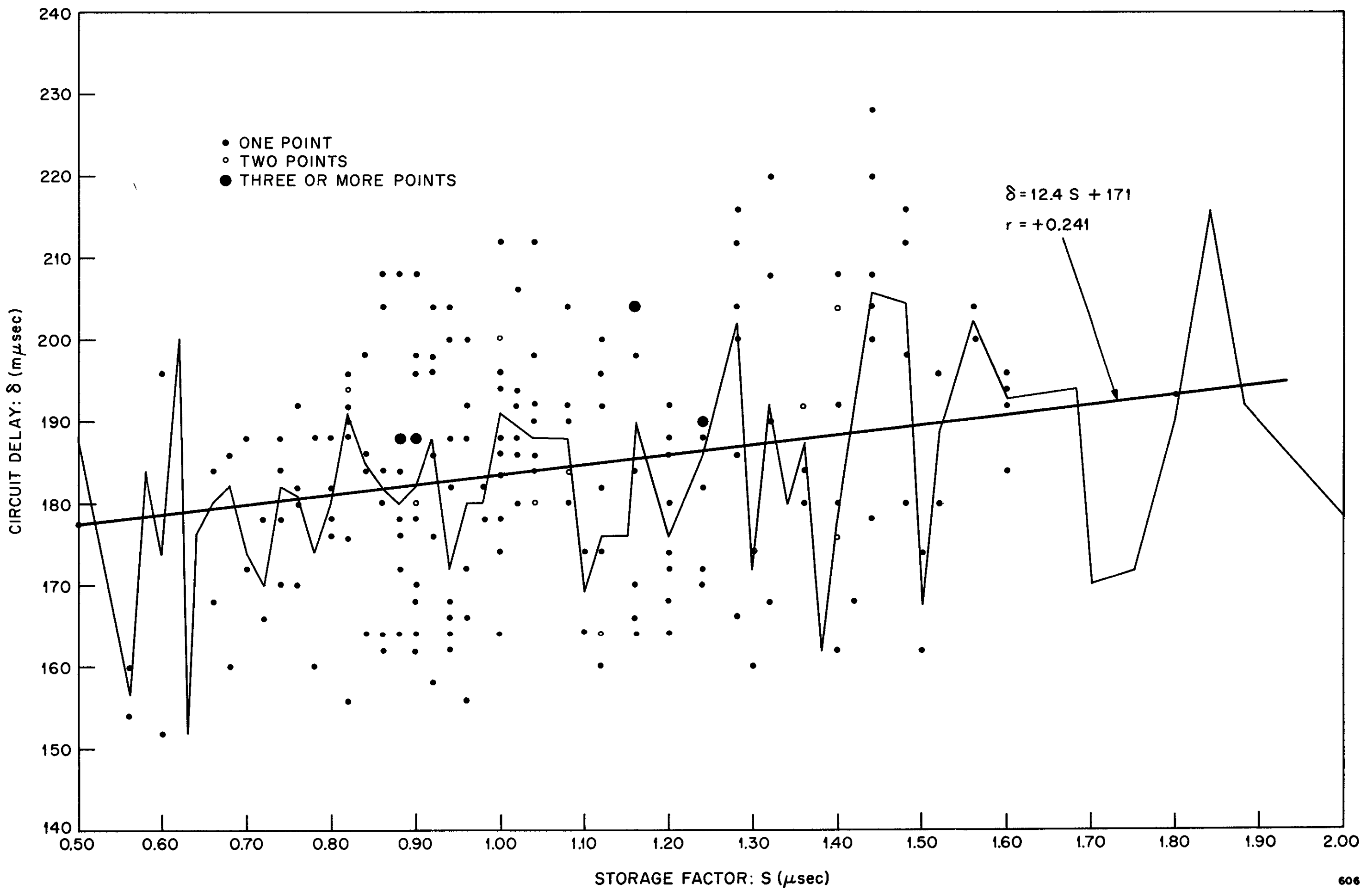


Figure 2-10. SCATTER DIAGRAM AND REGRESSION LINE FOR CIRCUIT DELAY AS A FUNCTION OF STORAGE FACTOR

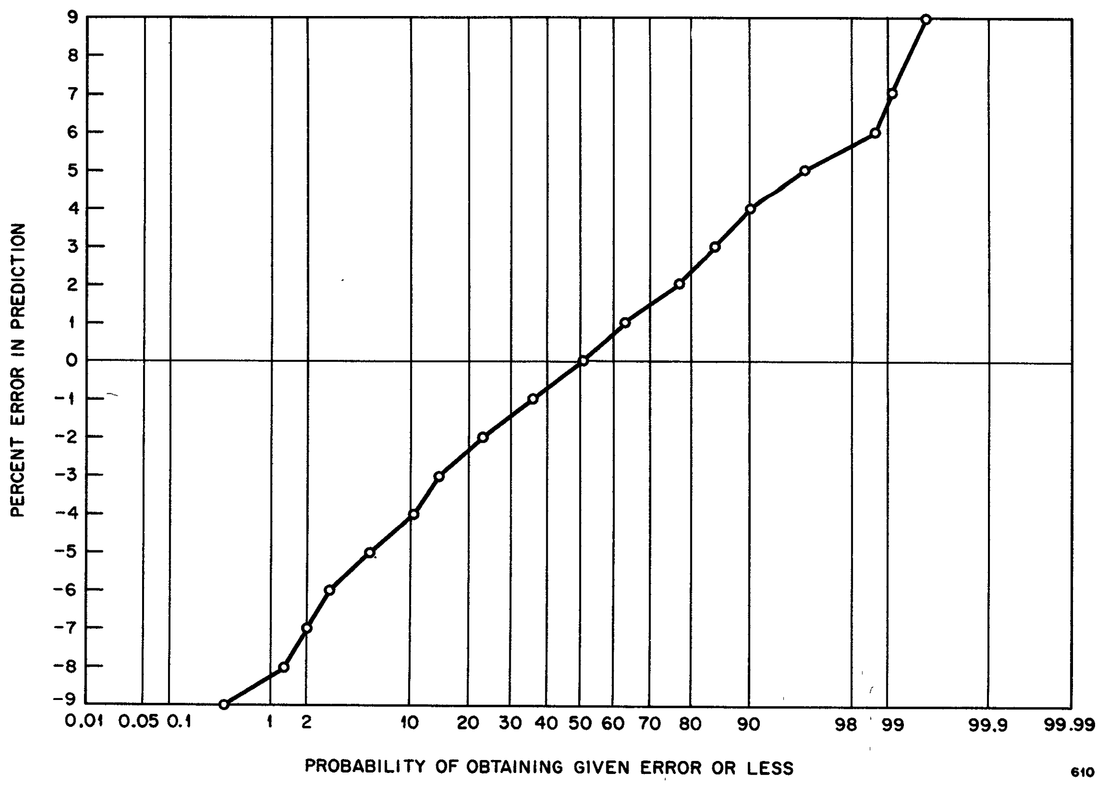


Figure 2-11. CIRCUIT DELAY PREDICTION, 234GT 762 TRANSISTORS TEST FOR NORMALITY OF ERROR DISTRIBUTION

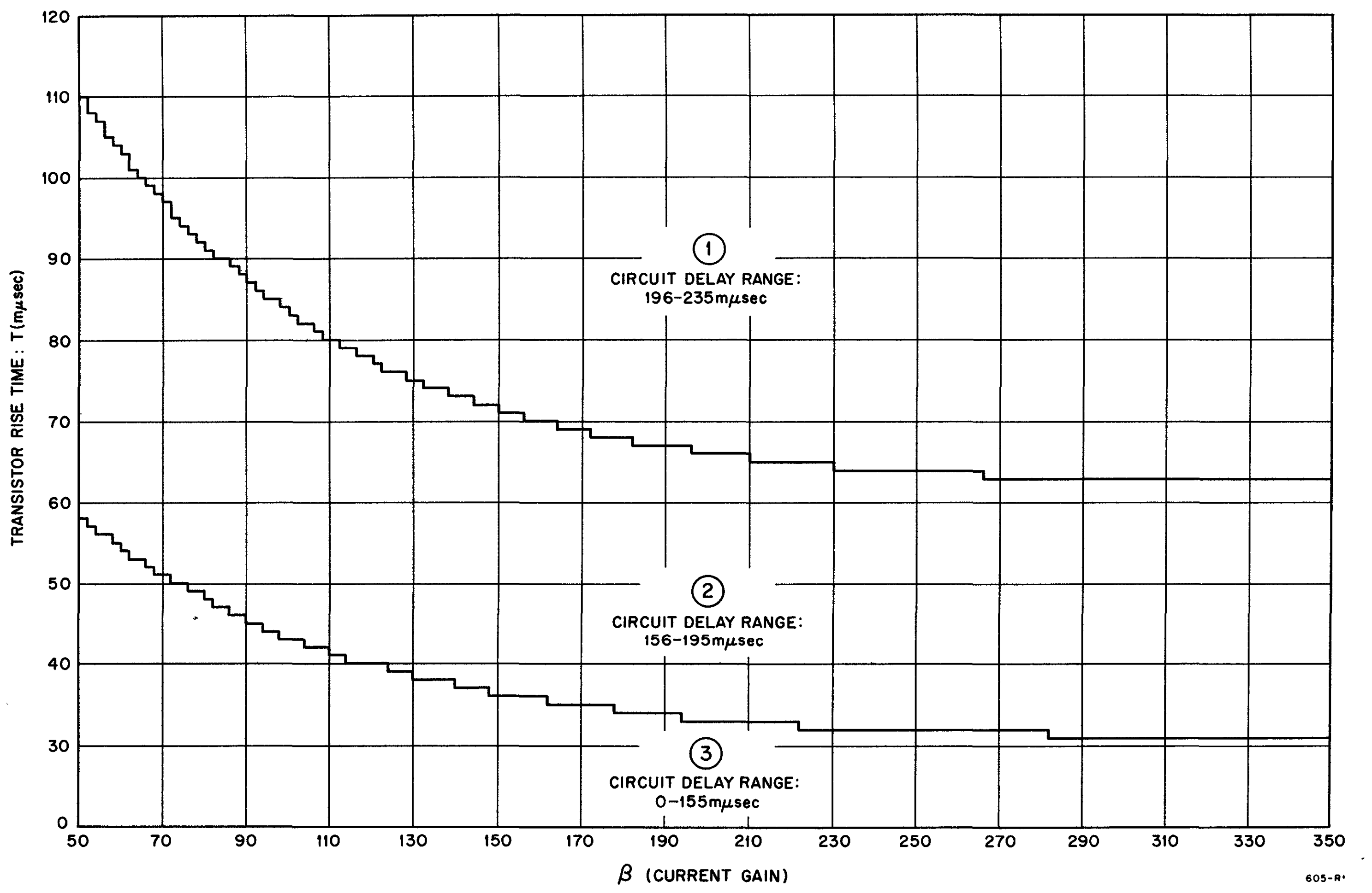


Figure 2-12. CIRCUIT DELAY RANGES

WIDE TEMPERATURE RANGE COINCIDENT CURRENT CORE MEMORIES

R. S. Weisz and M. Rosenberg
Ampex Computer Products Company
Culver City, California

A desired extension of man's intellect is to extremes of temperature where the human brain does not function effectively. Both military and industrial applications call for digital computers which can operate from polar cold to equatorial heat. In the space age, these limits must be broadened still further. Unfortunately, core memories are generally somewhat less tolerant of temperature changes than are human beings. Chiefly at fault is the ferrite core itself, because of its high temperature coefficient of coercivity, 0.5 to 0.7% per degree Centigrade. Nevertheless, since ferrite cores have proven to be highly reliable, reasonably fast, light in weight, and modest in power requirements, some means of overcoming the high temperature coefficient is needed to advance the art of computing.

A number of such schemes have been proposed. They include: (1) Compensating the drive circuits for the decrease in required current with rising temperature; (2) putting the memory in a thermostatically controlled chamber above the highest ambient anticipated; (3) using multi-apertured ferrite cores to provide a wide margin of current tolerance; (4) using a "word select" or linear select mode of address; (5) using metal alloys which have inherently lower temperature coefficients. Each of these methods may have some merit; certainly all have serious limitations.

Clearly, what one would like is a ferrite core having temperature independent properties. We have recently developed such a core. Aside from a low temperature coefficient of coercivity, the new core has a better disturb ratio than that of standard ferrite memory cores over the range -55° to +100°C. Valuable for other purposes as well, a high disturb ratio also aids in decreasing temperature sensitivity. An array of 1092 bits has been built from these cores using a simple toroidal geometry. It has been successfully operated over the range -55° to +125°C in a coincident current mode without temperature or current compensation.

In the remainder of this paper, we shall discuss in further detail the problem of wide temperature memory operation, the prior art on the subject, the new ferrite's characteristics, and the new memory's operation.

Problem of Wide Temperature Operation

To match commonly called for military specifications, we shall discuss memory operation over the range -55° to +125°C. It is important to note that there already exist components other than memory cores which can be operated over this same range. For example, transistors, diodes, resistors and capacitors, all necessary in memory circuits, are available.

A fundamental problem involved in operating a memory core over a wide temperature range will be discussed in reference to Figure 1, which is a superposition of hysteresis loops taken at -55°, +25°, and +100°C for a conventional memory core. It is readily seen that a current which is sufficient to switch the core at -55° is more than twice the required current at 25°C. Therefore, with the usual coincident current scheme, if half such a current is applied at room temperature, it will completely switch what are intended to be half-selected (unswitched) cores. Conversely, at low temperatures of operation, it is impossible to switch a core using the smaller current required at higher temperatures.

In any usable memory core, the ratio of the threshold to the full drive (or disturb ratio, R_d, as it is commonly called) is actually greater than 0.5. The resulting current tolerance can be used to provide a margin against drift in the drive circuits or changes in temperature. Conventional type memory cores are known with R_d as high as 0.67. If no current drift need be provided for, such a core can be operated over a range of approximately 50°C. A more realistic view is to allow for a drift of ±10% in drive current leaving an operating range of only 20°C.

One of the first methods suggested for extending the temperature range of operation to the desired 180°C was to provide compensation in the drive circuitry to decrease the current with increasing temperature so as to match the decreasing coercive force. Over a large temperature range a close match is difficult, but perhaps possible. A more serious problem, however, is also present. Those cores which are being switched during a temperature change will obviously settle on the appropriate new hysteresis loop as shown in Figure 1. But what happens to a non-selected core? In the absence of an applied field, it is believed that it remains at or close to its original remanent point. Thus, at the end of the temperature change there will be cores on two different hysteresis loops. It follows that spurious signals would then be obtained upon interrogating the memory. Despite this limitation, current compensation is successfully used over a narrower temperature range providing operation from 0° to 60°C.

Another approach that avoids the above problems completely, is to place the memory in a container which is kept at a controlled temperature above the highest anticipated.[1] Although this scheme has the merit of simplicity, it sacrifices some reliability due to the possibility of failure in the heater circuits. There is a further limitation inherent in the power requirement of the heaters. For example, to heat a memory of 3200 bits from -55° to +125°C, 10 watts are required.[1] A power level of this magnitude is far beyond the supply available in satellites now and in the predictable future; it is also an order of magnitude above the power requirements of the memory itself.

Still another partially successful approach is to use multi-apertured cores or "transfluxors."[2] Here one makes use of the geometry to provide a wider current tolerance than is possible with simple cores. Memories of this type have been operated over the range -20° to +100°C with non-destructive readout. The principal disadvantages of transfluxor memories are: (1) Their complicated wiring, and (2) increased use of semi-conductors. The cores are also much larger than simple toroids.

Another approach has been to use ferrite cores in a word select mode of operation.[3] This has the disadvantage of using a greater number of semi-conductors, resulting in decreased reliability and increased costs.

Finally, it has been proposed to abandon the square loop ferrites entirely and substitute metals or alloys in various configurations.[4,5] Permalloy and other nickel-iron alloys have very low temperature coefficients of coercivity. The principal disadvantage is their high electrical conductivity. One is forced to use very thin film to avoid eddy-current losses and as a result, two new problems arise. First, thin films are not well suited to wide temperature cycling because a difference in coefficient of expansion between the film and substrate causes cumulative stresses to be set up. Second, metalic films are prone to destructive oxidation at elevated temperatures, especially in the presence of moisture. None of the thin metal film devices has yet found wide acceptance. It is believed that uniformity of individual storage elements has been a more serious problem here than with discrete ferrite toroids. Furthermore, these elements have to date been operated only in word select type memories. This results once again in the use of a greater number of semi-conductors than would be used in an equivalent size coincident current memory.

New Ferrite Cores

Since none of the previously proposed methods of wide temperature memory operation seemed to be free of serious limitations, we embarked on a program to develop a square loop ferrite with a low temperature coefficient of coercivity. A family of such materials has now been found. Fortuitously, some of these also have a disturb ratio higher than conventional cores over the range -55° to +100°C. Details of the chemistry and physics of the new core are outside the scope of this article, but will be given in another publication.[6] A particular core was chosen for the memory to be described on the basis of optimum squareness, temperature coefficient, and speed. The dimensions of the toroid are standard: 50 mils O.D. x 30 mils I.D. x 15 mils thick.

Important characteristics of the cores are given in Figures 2 through 4. Figure 2 shows a superposition of 71 kc hysteresis loops of the wide temperature range core taken at -55°, +25°, and +100°C. For comparison purposes, Figure 1 shows the corresponding loops of a conventional magnesium-manganese ferrite core with the lowest temperature coefficient we have found among the presently used materials. One will observe that the new core has a much lower temperature coefficient of coercive force (approximately 0.13% per degree Centigrade as compared with 0.5% per degree Centigrade for the magnesium-manganese ferrite). The usual decrease in saturation flux density with increasing temperature is also much lower for the new core. Undoubtedly, this is connected with a high Curie temperature (greater than 500°C vs. 300°C or less for the magnesium-manganese ferrites).

More striking, and more to the point, are the pulse characteristics of the new core. Figure 3 epitomizes this data, giving the following characteristics as a function of temperature for a constant drive of 1.0 ampere turn: Switching time, t_s, and peaking time, t_p, in microseconds; undisturbed output uV_1, and noise, dV_z, in millivolts; disturb ratio, R_d. This data was obtained with a pulse rise time,

t_r, of 0.2 microseconds, a pulse width, t_d, of 10 microseconds, and 20 repeats of the disturbing current, whose magnitude was 0.5 ampere turn. The most striking feature is simply that a core can be operated over the range -55° to +100°C with a constant current. As stated previously, conventional cores at a constant drive cannot be operated over more than a 50°C spread.

Also important is the disturb ratio which is greater than 0.69 up to 100°C. Conventional cores have R_d up to 0.67. With R_d equal to 0.76 at room temperature, it is possible to operate the new core easily on a triple coincidence scheme. Although this possibility cannot be exploited simultaneously with the wide temperature operation, it is a promising lead for future work.

Figure 4 gives more detailed information on the core characteristics as a function of variable drive current. Other parameters of the drive pulses were made the same as for Figure 3. It should be stated at this time that these figures show tentative values for early cores. Recent improvements have lowered the noise voltage from that shown with no deleterious effects on other parameters. Signal-to-noise ratios of 10:1 without time strobing are now consistently obtained at room temperature under normal drive conditions.

Since high reliability is one of the chief attributes of ferrite memory cores, we felt that a life test of the new type at an elevated temperature was essential. In one such test, cores have been aged in air at 100°C for over eight months. Periodic test has shown no significant changes in pulse properties. In another test, cores were aged in a vacuum at 10^{-4} millimeters of Hg at 100°C for three months. At the end of this period, the samples were returned to room temperature conditions and retested. Again, no significant change was observed.

Besides the 50-mil diameter toroids, sample quantities of a 30-mil toroid, an 80-mil toroid, and a multi-apertured structure have been made of the new material. All have shown similar temperature characteristics.

Memory Characteristics

Having established the unusual temperature characteristics of the new core, system evaluation was carried out with the construction and test of a model memory array. It was felt that this was an essential part of the evaluation since, at times, components which have appeared to be usable in individual tests have failed when operated in systems. Memory elements in particular are likely to show interactions, delta noise problems, etc.

A standard type of coincident current array with 40 x 30 cores was constructed for the test. With a constant, uncompensated drive, the array was successfully operated in a temperature test chamber from -55° to +125°C. Figure 5 shows the direct output from the sense winding under worst pattern conditions at +100°C; Figure 6 shows the direct output from the sense winding at -55°C. The pulse conditions in both cases are as follows:

Pulse width = 3 μs
Pulse rise time = 0.75 μs
Repetition rate = 50 kc

The signal-to-noise ratio at peaking time of the "one" signal was greater than 50:1 in both cases. The variation in sense amplitude output over the temperature range is less than 2:1 (90 millivolts to 60 millivolts). This change is well within the dynamic range of a properly designed sense amplifier, particularly in view of the excellent signal-to-noise ratio.

Since the cores nominally switch in a microsecond, they can be used in a 5-6 microsecond memory system.

During test, the array was operated in a 3:2 selection mode by using 0.667 NI in the X lines and 0.333 NI in the Y lines. At 25°C and worst pattern conditions, a signal-to-noise ratio of better than 10:1 at peaking time was obtained. This simple experiment indicates that it is possible to operate in a true three-dimensional selection system. Further work is planned along these lines.

Conclusions

It is believed that the described array is a prototype for the first true wide temperature range memory. The well-known resistance of ferrites to oxidation, corrosion, and radiation damage can also be used to advantage in adverse environments. At the present time, a memory for a satellite is being built with the new type core. This memory has approximately one thousand bits operating in a coincident current mode over a temperature range of -55° to +100°C without compensation of current or temperature. Other interesting properties and applications of the new memory core will be discussed in later papers.

References

1. Miniature Memory Plane for Extreme Environmental Conditions, R. Straley, A. Heuer, B. Kane, and G. Tkach. Journal of Applied Physics, vol. 31, April 1960, pp. 126s - 128s.

2. Temperature Characteristics of the Transfluxor, H. W. Abbott and J. J. Suran. IRE Transactions on Electron Devices, vol. ED-4, April 1957, pp. 113-119.

3. Design of a Reliable High Speed Militarized Core Memory, M. Stern and H. Ullman. Program of the Winter Conference on Military Electronics (IRE), Los Angeles, Feb. 1, 1961. (in abstract form)

4. Recent Advances in Magnetic Devices for Computers, D. H. Looney. Journal of Applied Physics, vol. 30, April 1959, pp. 38s-42s.

5. Millimicrosecond Magnetic Switching and Storage Element, D. A. Meier, Journal of Applied Physics, vol. 30, April 1959, pp. 45s-46s.

6. Square Loop Ferrites With Temperature Independent Properties and Improved Disturb Ratio, R. S. Weisz. Accepted for publication in Journal of Applied Physics.

Captions

Figure 1 - Hysteresis Loops of Standard One Microsecond Memory Core at -55°, +25°, and +100°C. Vertical Scale = 680 gauss/div. Horizontal Scale = 0.78 oe/div.

Figure 2 - Hysteresis Loops of Wide Temperature Range Core at -55°C, +25°C, and +100°C. Vertical Scale = 940 gauss/div. Horizontal Scale = 1.4 oe/div.

Figure 3 - Pulse Characteristics of Wide Temperature Range Core as a Function of Temperature at a Constant Drive of 1.0 Ampere Turn.

Figure 4 - Pulse Characteristics of Wide Temperature Range Core as a Function of Drive.

Figure 5 - Array Output at 100°C. Vertical Scale = 50 mv/div. Horizontal Scale = 0.5 μs/div.

Figure 6 - Array Output at -55°C. Vertical Scale = 50 mv/div. Horizontal Scale = 0.5 μs/div.

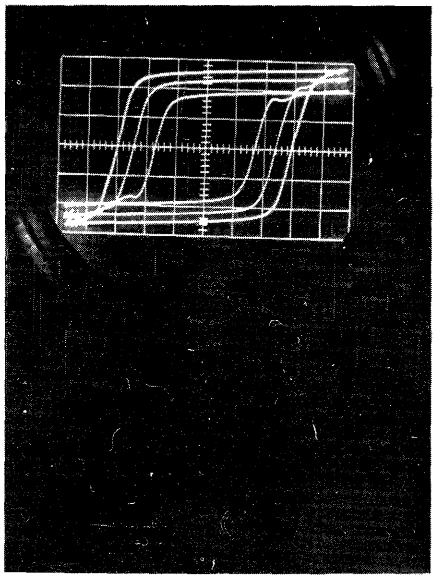


Figure 1. HYSTERESIS LOOPS OF STANDARD ONE MICROSECOND MEMORY CORE AT -55°, ≠25°, AND ≠100° C.
Vertical Scale = 680 gauss/div.
Horizontal Scale = 0.78 oe/div.

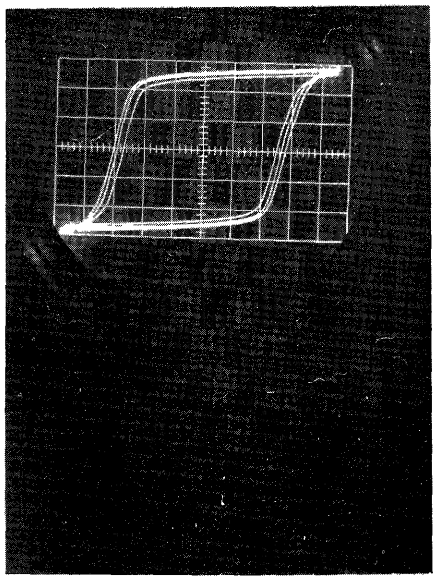


Figure 2. HYSTERESIS LOOPS OF WIDE TEMPERATURE RANGE CORE AT -55°C, ≠25°C, AND ≠100°C.
Vertical Scale = 940 gauss/div.
Horizontal Scale = 1.4 oe/div.

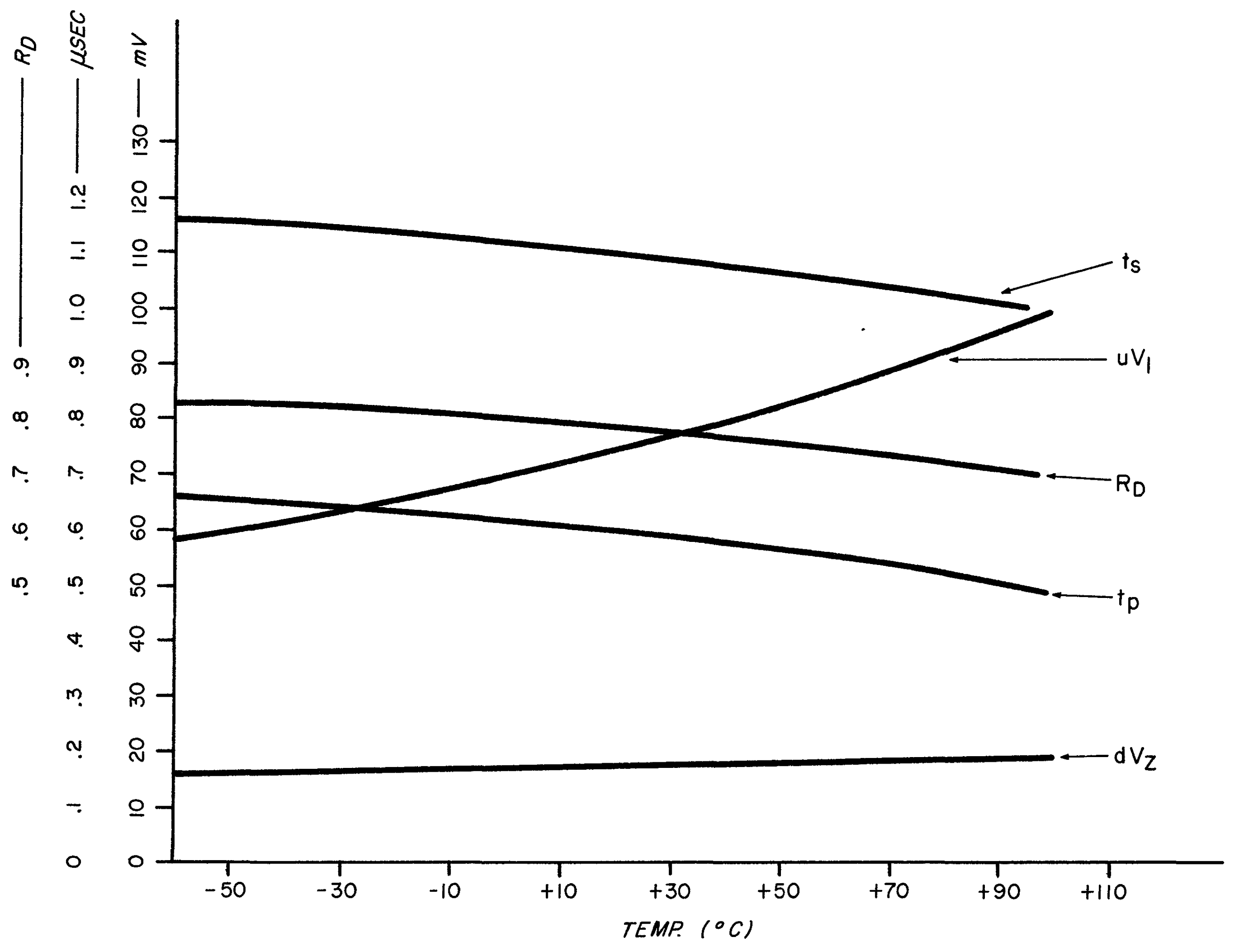


Figure 3. PULSE CHARACTERISTICS OF WIDE TEMPERATURE RANGE CORE AS A FUNCTION OF TEMPERATURE
Constant Drive of 1.0 Ampere Turn

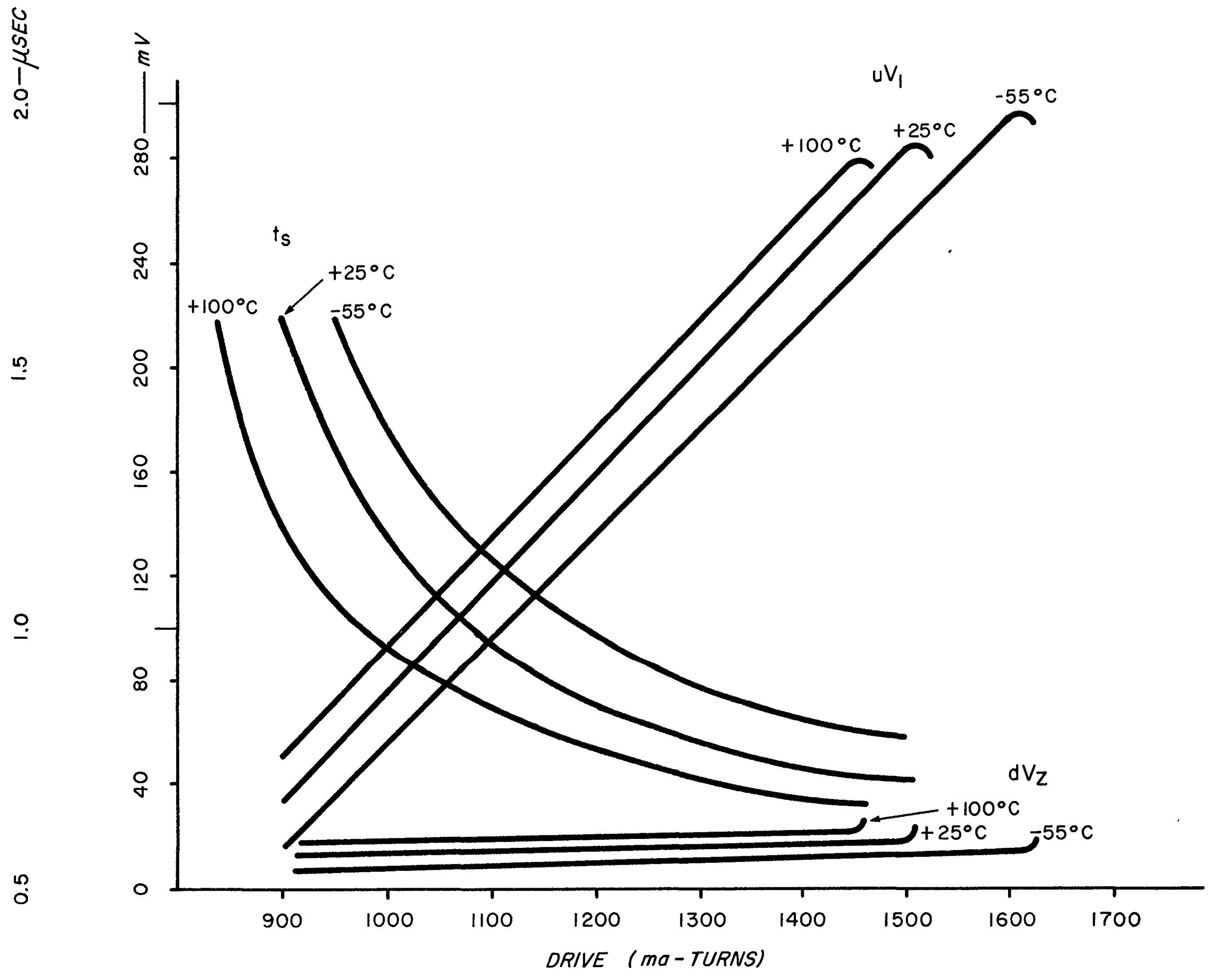


Figure 4. PULSE CHARACTERISTICS OF WIDE TEMPERATURE RANGE CORE AS A FUNCTION OF DRIVE

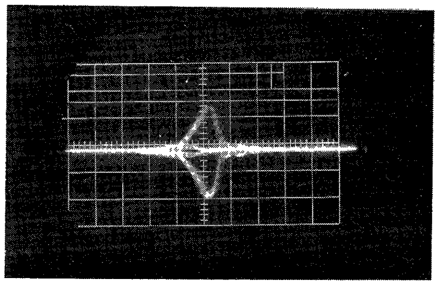


Figure 5. ARRAY OUTPUT AT 100°C.
Vertical Scale = 50 mv/div.
Horizontal Scale = 0.5 μ s/div.

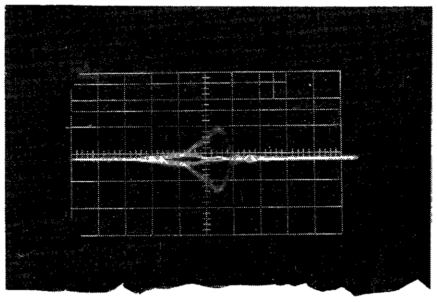


Figure 6. ARRAY OUTPUT AT -55°C.
Vertical Scale = 50 mv/div.
Horizontal Scale = 0.5 μ s/div.

DESCRIPTIVE LANGUAGES AND PROBLEM SOLVING

Marvin Minsky
Dept. of Mathematics and Computation Center
Massachusetts Institute of Technology

Advances in machine problem solving may depend on use of internal languages for description and abstraction of the outcomes of experiments. As more complex problems are attempted there will have to be less trial and error and more systematic analysis of the results of each trial. Learning on the basis of experience will require a phase of refinement in which the machine will attempt, by analysis and inductive inference, to get as much as possible from each experiment.

Introduction

Work on artificial intelligence is proceeding at a slow, apparently steady, rate. The complexity of problems being attacked is growing slowly, as is the complexity of the successful programs themselves. In the past it seems to have taken two or three years for each significant advance and one may ask why progress is so slow. Much of this time has been spent on the development of programming languages and systems suitable for the symbol manipulation processes involved. But much of the difficulty has been conceptual also. The methods which worked quite well on easy problems did not extend smoothly to the difficult ones. Continued progress will require implementation of new ideas, for there are some very tough problems in our immediate path. It seems to us that solution of these problems will require the use of non-trivial formal and descriptive language systems. These are only beginning to appear as a working part of the problem solving machinery and it will take much ingenuity to bring current notions into usable form.

The two papers of this session represent important, and very different, phases in the development of machine-usable language systems. In one case we have a system which can process, to find the meaning with respect to a small universe, expressions in a very life-like fragment of ordinary language.

In the second paper we find an ambitious attempt at the beginnings of a Theory of Computation, based in part on the use of a symbol manipulation language suited at once for both theoretical analysis and for practical programming use.

Our purpose here is to indicate a few of the considerations that seem to point toward the incorporation of complex linguistic processes into the next generation of heuristic programs. Some of these difficulties have arisen in the author's work, jointly with McCarthy, on the Advice-Taker system[1]. In a recent paper[2] the author discussed the principles and mechanisms of a variety of problem solving systems, but did not dwell on the question of extending these to really complex problems. We assume the terminology of that paper. When one attempts to apply the techniques described there one discovers that

1. The search problems become very serious. One is faced not only with greatly enlarged problem trees but also with a greater variety of plausible methods.

2. The problem of learning from experience becomes qualitatively more difficult. To learn the lesson of a complex experience requires shrewd, deliberate, analysis that cannot be approximated by any of the simple learning models based on averaging or on correlation.

3. The classification and pattern recognition methods must be on a descriptive level. Again, correlation or matching methods must be replaced by more sophisticated symbol-manipulation processes.

4. Planning methods, Character and Difference algebras, etc., threaten to collapse when the fixed sets of categories adequate for simple problems have to be replaced by the expressions of a descriptive language. The use of look-up tables for choosing methods will have to be supplemented by something more like reasoning.

When we call for the use of "reasoning" we intend no suggestion of giving up the game by invoking an intelligent subroutine. The program that administers the search will be just another heuristic program. Almost certainly it will be composed largely of the same sorts of objects and processes that will comprise the subject-domain programs. Almost certainly it will be recursively applied to itself so that the system can be finite. But it does seem clear that the basic (non-recursive) part of the structure will have to be more complex than is any current system.

The Need for Analysis

The simplest problems, e.g., playing tic-tac-toe or proving the very simplest theorems of logic, can be solved by simple recursive application of all the available transformations to all the situations that occur, dealing with subproblems in the order of their generation. This becomes impractical in more complex problems as the search space grows larger and each trial becomes more expensive in time and effort. One can no longer afford a policy of simply leaving one unsuccessful attempt to go on to another. For each attempt on a difficult problem will involve so much effort that one must be quite sure that, whatever the outcome, the effort will not be wasted entirely. One must become selective to the point that no trial is made without a compelling reason; just as in any research, expensive experiments must be carefully designed. One must do a good deal of criticism and analysis between experiments so that each will be a critical test of a significant portion of the search space.

The ability to solve a difficult problem hinges on the ability to split or transform it into problems of a lower order of difficulty. To do this, without total reliance on luck, requires some understanding of the situation. One must be able to deduce, or guess, enough of the consequences of the problem statement to be able to set up simpler models of the problem situation. The models must have enough structure to make it likely that there will be a way to extend their solutions to the original problem.

The construction of less difficult subproblems will be useful, by definition, only if one has already a very good chance of solving _them_ efficiently. Otherwise the search tree will grow beyond bounds. This means we must have already built up adequate solution methods for the lower order problems, e.g., as a set of more or less packaged subroutines. This entails some formidable requirements:

Training Sequences

The machine is presumed to have acquired its good subroutines through earlier solution of less complex problems. (We are not interested here in the case in which these methods are provided at the start.) Thus the machine must have been exposed to a graded sequence of problems. To be sure, given time-limits, a machine will select a graded subsequence from an unorganized variety of problems. But a careful arrangement will be necessary to insure that methods learned in the problems that the machine does manage to solve will be useful on more difficult problems met later. In any case one cannot rely on making large jumps, either in machines or in humans.

Refinement Phase

Solving simpler problems is not enough. To make progress one needs also to "package" the successful method for effective later use. We are not interested in the trivial case of recognizing a problem once before solved, though this can be difficult enough when there is some disguise. The success must be generalized to cover a substantial variety of situations. To do this it would seem that there should be a phase of exploration and consolidation in which the successful method is refined--its central innovation (if any) isolated and packaged, in terms as general as possible. One must explore its range of application and construct an expression describing this range. This may involve inventing similar problems on which the method, or close variant, works; then constructing a plausible generalization.

Certainly people must go through such phases. One cannot usually solve hard problems with once-used but still unfamiliar methods. One must first "understand" the methods quite well; this means becoming able to recognize situations in which they are applicable. It is probably misleading to think of this as "practice"--acquisition of facility through repetition. Exercise in, e.g., mathematical technique is probably very different from exercise in weight-lifting. Its effect is not so much in reinforcing methods, or paths already weakly laid down, but is rather to provide the necessary data for some Inductive Inference technique. The latter will replace the special method by one of somewhat greater generality.

Failure of the refinement phase to yield a precise, abstractly stated conclusion can be concealed to a point. One often encounters mathematical situations in which one can answer particular questions quickly, yet is unable to state a satisfactory formal generalization. This can happen through the assembly of a set of different models or examples which, as a group, show most or all of the features of the unformulated general theorem. One can answer some question in the negative, by finding inconsistency with an example. Consistency with all leads one to the affirmative. Often the examples themselves are not formulated clearly, or completely consciously. In such cases one will find some statements seem "obvious" yet (because of the incomplete understanding which precludes giving any precise explanation) are also felt to be "intuitive." An incomplete formalization or conceptualization, e.g., such a set of examples, can be very powerful when used at or near the top level. But if not understood or "packaged" it could become a serious nuisance later when, because of its informality, it cannot be used in deduction or in the construction of further abstractions.

Coding and Retrieval Problems

The compact representation of results of previous experience requires an adequate descriptive language. This language must permit general statements about both problem-domain matters and about the problem-solving methods. It must permit logical deductions to be made. This raises several problems.

One problem that has been a great nuisance to us arises in connection with non-mathematical problems in which actions affect the state of some subject domain. Thus a move affects the positions of pieces in a board game. When this happens, some statements formerly deduced about the situation cease to be true. (In a mathematical domain a theorem, once proved, remains true when one proves other theorems!) One must then deduce all the consequences of an action in so far as it affects propositions that one is planning to use. This might be done through some heuristic technique which can assess relevancy, or it could be done through a logic which takes such consequences into account. The trouble with the latter is that the antecedents of all the propositions must contain a condition about the state of the system, and for complex systems this becomes overwhelmingly cumbersome. Other systematic solutions to the problem seem about equally repellent. It is a problem that seems urgently to require a heuristic solution.

Our present proposal on this matter is to make the system plan ahead. Whenever an important deduction is made, the system is to try to discover which kinds of actions could affect its validity. Independent monitors are then set up to detect when such actions are proposed. The normal problem solving exploration process proceeds independently of these monitors, and is interrupted when one of them detects a threat to the proposition it is defending. This model has a certain introspectively attractive character; it suggests a free conscious exploration with more or less subconscious trouble-detectors. Unfortunately, its essentially parallel nature threatens to make its use in serial computer programming rather expensive. We hope someone will come up with a better idea.

In any case, the retrieval problem has to be faced. The problem of making useful deductions from a large body of statements (e.g., about the relevance of different methods to different kinds of problems) raises a new search problem. One must restrict the logical exploration to data likely to be relevant to the current problem. This selection function could hardly be completely built-in at the start. It must develop along with other data accumulated by experience.

Another rather serious problem centers around the problem of abbreviations, or proper names. The language must be used together with an abbreviative technique so that the most useful notions can be designated by reasonably convenient (short) representations. This is not only a matter of convenience and compactness; it is a more or less inescapable requirement of known inductive inference techniques and thus requisite for formation of hypotheses or generalizations. Unfortunately an abbreviation cannot show all of the structure of the longer expression it designates. This seriously limits the possibilities of making formal logical deductions. Ultimately the machines will have to use mnemonic codings in their internal languages, just as we need to do this when we use their external languages.

The systematic solution to the abbreviation problem is, again, to revise the whole body of propositions in current use, in so far as they are going to be used in the same deductive operations. All the alternatives to this that we can envision are of somewhat stopgap nature. We content ourselves with the observation that it is equally a major problem for humans to make substantial changes in basic abstractions, ways of classifying or perceiving, and the like. Once one has built up a structure depending on a certain conceptual commitment, he will stave off a revision of its foundation as though the cost of changing it were high. Otherwise, perhaps, people would not argue so much. One may view that phenomenon, if one likes, as a matter of ego involvement. But it would be well to remember that being wrong (and having to change) has a real intellectual cost, and not merely a social cost.

In any case, our ideas on this subject are not yet in presentable condition.

Conclusion

The need to be able to make abstractions in symbolic language is already urgent in current attempts to make machines prove theorems, play games, etc. There are some very difficult problems to be faced in this area. We still maintain, with McCarthy, that "in order for a program to be capable of learning something, it must first be capable of being told it."[1] Results on "self-organizing systems" without explicit provision for such abilities show very little promise to date, and systematic attempts in the direction of internal language processing should be promoted.

References

1. J. McCarthy, "Programs with Common Sense," Mechanization of Thought Processes, H.M.S.O., London, 1959, pp. 75-84 (Vol. I).

2. M. L. Minsky, "Steps Toward Artificial Intelligence," Proc. IRE, Vol. 49, no. 1, Jan. 1961, pp. 8-30.

BASEBALL: AN AUTOMATIC QUESTION-ANSWERER

Bert F. Green, Jr., Alice K. Wolf, Carol Chomsky, and Kenneth Laughery
Lincoln Laboratory*, Massachusetts Institute of Technology
Lexington 73, Massachusetts

Summary

Baseball is a computer program that answers questions phrased in ordinary English about stored data. The program reads the question from punched cards. After the words and idioms are looked up in a dictionary, the phrase structure and other syntactic facts are determined for a content analysis, which lists attribute-value pairs specifying the information given and the information requested. The requested information is then extracted from the data matching the specifications, and any necessary processing is done. Finally, the answer is printed. The program's present context is baseball games; it answers such questions as "Where did each team play on July 7?"

Introduction

Men typically communicate with computers in a variety of artificial, stylized, unambiguous languages that are better adapted to the machine than to the man. For convenience and speed, many future computer-centered systems will require men to communicate with computers in natural language. The business executive, the military commander, and the scientist need to ask questions of the computer in ordinary English, and to have the computer answer questions directly. Baseball is a first step toward this goal.

Baseball is a computer program that answers questions posed in ordinary English about data in its store. The program consists of two parts. The linguistic part reads the question from a punched card, analyzes it syntactically, and determines what information is given about the data being requested. The processor searches through the data for the appropriate information, processes the results of the search, and prints the answer.

The program is written in IPL-V[1], an information processing language that uses lists, and hierarchies of lists, called list structures, to represent information. Both the data and the dictionary are list structures, in which items of information are expressed as attribute-value pairs, e.g., Team = Red Sox.

*Operated with support from the U.S. Army, Navy, and Air Force.

The program operates in the context of baseball data. At present, the data are the month, day, place, teams and scores for each game in the American League for one year. In this limited context, a small vocabulary is sufficient, the data are simple, and the subject-matter is familiar.

Some temporary restrictions were placed on the input questions so that the initial program could be relatively straightforward. Questions are limited to a single clause; by prohibiting structures with dependent clauses the syntactic analysis is considerably simplified. Logical connectives, such as and, or, and not, are prohibited, as are constructions implying relations like most and highest. Finally, questions involving sequential facts, such as "Did the Red Sox ever win six games in a row?" are prohibited. These restrictions are temporary expedients that will be removed in later versions of the program. Moreover, they do not seriously reduce the number of questions that the program is capable of answering. From simple questions such as "Who did the Red Sox lose to on July 5?" to complex questions such as "Did every team play at least once in each park in each month?" lies a vast number of answerable questions.

Specification List

Fundamental to the operation of the baseball program is the concept of the specification list, or spec list. This list can be viewed as a canonical expression for the meaning of the question; it represents the information contained in the question in the form of attribute-value pairs, e.g., Team = Red Sox. The spec list is generated from the question by the linguistic part of the program, and it governs the operation of the processor. For example, the question "Where did the Red Sox play on July 7?" has the spec list:

```
Place = ?
Team  = Red Sox
Month = July
Day   = 7
```

Some questions cannot be expressed solely in terms of the main attributes (Month, Day, Place, Team, Score and Game Serial Number), but require some modification of these attributes. For example, on the spec list of "What teams won 10 games in July?", the attribute Team is modified by Winning, and Game is modified by Number of, yielding

$$\text{Team}_{(\text{winning})} = ?$$

$$\text{Game}_{(\text{number of})} = 10$$

$$\text{Month} = \text{July}$$

Dictionary

The dictionary definitions, which are expressed as attribute-value pairs, are used by the linguistic part of the program in generating the spec list. A complete definition for a word or idiom includes a part of speech, for use in determining phrase structure; a meaning, for use in analyzing content; an indication of whether the entry is a question-word, e.g., who or how many; and an indication of whether a word occurs as part of any stored idiom. Separate dictionaries are kept for words and idioms, an idiom being any contiguous set of words that functions as a unit, having a unique definition.

The meaning of a word can take one of several forms. It may be a main or derived attribute with an associated value. For example, the meaning of the word Team is Team = (blank), the meaning of Red Sox is Team = Red Sox, and the meaning of who is Team = ?. The meaning may designate a subroutine, together with a particular value, as in the case of modifiers such as winning, any, six, or how many. For example, winning has the meaning Subroutine A1 = Winning. The subroutine, which is executed by the content analysis, attaches the modifier Winning to the attribute of the appropriate noun. Some words have more than one meaning; the word Boston may mean either Place = Boston or Team = Red Sox. The dictionary entry for such words contains, in addition to each meaning, the designation of a subroutine that selects the appropriate meaning according to the context in which the word is encounted. Finally, some words such as the, did, play, etc., have no meaning.

Data

The data are organized in a hierarchical structure, like an outline, with each level containing one or more items of information. Relationships among items are expressed by their occurrence on the same list, or on associated lists. The main heading, or highest level of the structure, is the attribute Month. For each month, the data are further subdivided by place. Below each place under each month is a list of all games played at that place during that month. The complete set of items for one game is found by tracing one path through the hierarchy, i.e. one list at each level. Each path contains values for each of six attributes, e.g.:

```
Month = July
   Place = Boston
      Day              = 7
      Game Serial No. = 96
      (Team = Red Sox, Score = 5)
      (Team = Yankees, Score = 3)
```

The parentheses indicate that each Team must be associated with its own score, which is done by placing them together on a sublist.

The processing routines are written to accept any organization of the data. In fact, they will accept a non-parallel organization in which, for example, the data might be as above for all games through July 31, and then organized by place, with month under place, for the rest of the season. The processing routines will also accept a one-level structure in which each game is a list of all attribute-value pairs for that game. The possibility of hierarchical organization was included for generality and potential efficiency.

Details of the Program

The program is organized into several successive, essentially independent routines, each operating on the output of its predecessor and producing an input for the routine that follows. The linguistic routines include question read-in, dictionary look-up, syntactic analysis, and content analysis. The processing routines include the processor and the responder.

Linguistic Routines

Question Read-in. A question for the program is read into the computer from punched cards. The question is formed into a sequential list of words.

Dictionary Look-up. Each word on the question list is looked up in the word dictionary and its definition copied. Any undefined words are printed out. (In the future, with a direct-entry keyboard, the computer can ask the questioner to define the unknown words in terms of words that it knows, and so augment its vocabulary.) The list is scanned for possible idioms; any contiguous words that form an idiom are replaced by a single entry on the question list, and an associated definition from the idiom dictionary. At this point, each entry on the list has associated with it a definition, including a part of speech, a meaning, and perhaps other indicators.

Syntax. The syntactic analysis is based on the parts of speech, which are syntactic categories assigned to words for use by the syntax routine. There are 14 parts of speech and several ambiguity markers.

First, the question is scanned for ambiguities in part of speech, which are resolved in some cases by looking at the adjoining words, and in other cases by inspecting the entire question. For example, the word score may be either a noun or a verb; our rule is that, if there is no other main verb in the question, then score is a verb, otherwise it is a noun.

Next, the syntactic routine locates and brackets the noun phrases, [ ] , and the prepositional and adverbial phrases, ( ). The verb is left unbracketed. This routine is patterned after the work of Harris and his associates at the University of Pennsylvania.[2] Bracketing proceeds from the end of the question to the beginning. Noun phrases, for example, are bracketed in the following manner: certain parts of speech indicate the end of a noun phrase; within a noun phrase, a part of speech may indicate that the word is within the phrase, or that the word starts the phrase, or that the word is not in the phrase, which means that the previous word started the phrase. Prepositional phrases consist of a preposition immediately preceding a noun phrase. The entire sequence, preposition and noun phrase, is enclosed in prepositional brackets. An example of a bracketed question is shown below:

[How many games] did

[the Yankees] play (in [July])?

When the question has been bracketed, any unbracketed preposition is attached to the first noun phrase in the sentence, and prepositional brackets added. For example, "Who did the Red Sox lose to on July 5?" becomes "(To [who]) did [the Red Sox] lose (on [July 5])?"

Following the phrase analysis, the syntax routine determines whether the verb is active or passive and locates its subject and object. Specifically, the verb is passive if and only if the last verb element in the question is a main verb and the preceding verb element is some form of the verb to be. For questions with active verbs, if a free noun phrase (one not enclosed in prepositional brackets) is found between two verb elements, it is marked Subject, and the first free noun phrase in the question is marked Object. Otherwise the first free noun phrase is the subject, the next, if any, is the object. For passive verbs, the first free noun phrase is marked Object (since it is the object in the active form of the question) and all prepositional phrases with the preposition by have the noun phrase within them marked Subject. If there is more than one, the content analysis later chooses among them on the basis of meaning.

Finally, the syntactic analysis checks to see if any of the words is marked as a question word. If not, a signal is set to indicate that the question requires a yes/no answer.

Content Analysis. The content analysis uses the dictionary meanings and the results of the syntactic analysis to set up a specification list for the processing program. First any subroutine found in the meaning of any word or idiom in the question is executed. The subroutines are of two basic types; those that deal with the meaning of the word itself and those that in some way change the meaning of another word. The first chooses the appropriate meaning for a word with multiple meanings, as, for example, the subroutine mentioned above that decides, for names of cities, whether the meaning is Team = A_t or Place = A_p. The second type alters or modifies the attribute or value of an appropriate syntactically related word. For example, one such subroutine puts its value in place of the value of the main noun in its phrase. Thus Team = (blank) in the phrase each team becomes Team = each; in the phrase what team, it becomes Team = ?. Another subroutine modifies the attribute of a main noun. Thus Team = (blank) in the phrase winning team becomes $\text{Team}_{(\text{winning})}$ = (blank). In the question "Who beat the Yankees on July 4?", this subroutine, found in the meaning of beat, modifies the attribute of the subject and object, so that Team = ? and Team = Yankees are rendered $\text{Team}_{(\text{winning})}$ = ? and $\text{Team}_{(\text{losing})}$ = Yankees. Another subroutine combines these two operations: it both modifies the attribute and changes the value of the main noun. Thus, Game = (blank) in the phrase six games becomes $\text{Game}_{(\text{number of})}$ = 6, and in the phrase how many games becomes $\text{Game}_{(\text{number of})}$ = ?.

After the subroutines have been executed, the question is scanned to consolidate those attribute-value pairs that must be represented on the specification list as a single entry. For example, in "Who was the winning team..." Team = ? and $\text{Team}_{(\text{winning})}$ = (blank) must be collapsed into $\text{Team}_{(\text{winning})}$ = ?. Next, successive scans will create any sublists implied by the syntactic structure of the question. Finally, the composite information for each phrase is entered onto the spec list. Depending on its complexity, each phrase furnishes one or more entries for the list. The resulting spec list is printed in outline form, to provide the questioner with some intermediate feedback.

Processing Routine

Processor. The specification list indicates to the processor what part of the stored data is relevant for answering the input question. The processor extracts the matching information from the data and produces, for the responder, the answer to the question in the form of a list structure.

The core of the processor is a search routine that attempts to find a match, on each path of a given data structure, for all the attribute-value pairs on the spec list; when a match for the whole spec list is found on a given path, these pairs relevant to the spec list are entered on a found list. A particular spec list pair is considered matched when its attribute has been found on a data path and, either the data value is the same as the spec value, or the spec value is ? or each, in which case any value of the particular attribute is a match. Matching is not always straightforward. Derived attributes and some modified attributes are functions of a number of attributes on a path and must be computed before the values can be matched. For example, if the spec entry is Home Team = Red Sox, the actual home team for a particular path must be computed from the place and teams on that path before the spec value Red Sox can be matched with the computed data value. Sublists also require special handling because the entries on the sublist must sometimes be considered separately and sometimes as a unit in various permutations.

The found list produced by the search routine is a hierarchical list structure containing one main or derived attribute on each level of each path. Each path on the found list represents the information extracted from one or more paths of the data. For example, for the question "Where did each team play in July?", a single path exists, on the found list, for each team which played in July. On the level below each team, all places in which that team played in July occur on a list that is the value of the attribute Place. Each path on the found list may thus represent a condensation of the information existing on many paths of the search data.

Many input questions contain only one query, as in the question above, i.e., Place = ?. These questions are answered, with no further processing, by the found list produced by one execution of the search routine. Others require simple processing on all occurrences of the queried attribute on the generated found list. The question "In how many places did each team play in July?" requires a count of the places for each team, after the search routine has generated the list of places for each team.

Other questions imply more than one search as well as additional processing. For a spec attribute with the value every, a comparison with a list of all possible values for that attribute must be made after the search routine has generated lists of found values for that attribute. Then, since only those found list paths for which all possible values of the attribute exist should remain on the found list as the answer to the question, the search routine, operating on this found list as the data, is again executed. It now generates a new found list containing all the data paths for which all possible values of the attribute were found. Likewise, questions involving a specified number, such as 4 teams, imply a search for which teams, a count of the teams found on each path, and a search of the found list for paths containing 4 teams.

In general, a question may contain implicit or explicit queries. Since these queries must be answered one at a time, several searches, with intermediate processing, are required. The first search operates on the stored data while successive searches operate on the found list generated by the preceding search operation. As an example, consider the question "On how many days in July did eight teams play?" The spec list is

Day (number of) = ? ;
Month = July;
Team (number of) = 8 .

On the first pass, the implicit question which teams is answered. The spec list for the first search is

Day = Each;
Month = July;
Team = ? .

The found data is a list of days in July; for each day there is a list of teams that played on that date. Following this search, the processor counts the teams for each day and associates the count with the attribute Team. On the second search, the spec list is

Day = ? ;
Month = July;
Team (number of) = 8 .

The found data is a list of days in July on which eight teams played. After this pass, the processor counts the days, addsthe count to the found list and is finished.

Responder. No attempt has yet been made to respond in grammatical English sentences. Instead, the final found list is printed, in outline form. For questions requiring a yes/no answer, YES is printed along with the found list. If the search routine found no matching data, NO is printed for yes/no questions, and NO DATA for all other cases.

Discussion

The differences between Baseball and both automatic language translation and information retrieval should now be evident. The linguistic part of the baseball program has as its main goal the understanding of the meaning of the question as embodied in the canonical specification list. Syntax must be considered and ambiguities resolved in order to represent the meaning adequately. Translation programs have a different goal: transforming the input passage from one natural language to another. Meanings must be considered and ambiguities resolved to the extent that they effect the correctness of the final translation. In general, translation programs are concerned more with syntax and less with meaning than the Baseball program.

Baseball differs from most retrieval systems in the nature of its data. Generally the retrieval problem is to locate relevant documents. Each document has an associated set of index numbers describing its content. The retrieval system must find the appropriate index numbers for each input request and then search for all documents bearing those index numbers. The basic problem in such systems is the assignment of index categories. In Baseball, on the other hand, the attributes of the data are very well specified. There is no confusion about them. However, Baseball's derived attributes and modifiers imply a great deal more data processing than most document retrieval programs. (Baseball does bear a close relation with the ACSI-MATIC system discussed by Miller et al at the 1960 Western Joint Computer Conference.[3])

The concept of the spec list can be used to define the class of questions that the baseball program can answer. It can answer all questions whose spec list consists of attribute-value pairs that the program recognizes. The attributes may be modified or derived, and the values may be definite or queries. Any combination of attribute-value pairs constitutes a specification list. Many will be nonsense, but all can be answered. The number of questions in the class is, of course, infinite, because of the numerical values. But even if all numbers are restricted to two digits, the program can answer millions of meaningful questions.

The present program, despite its restrictions, is a very useful communication device. Any complex question that does not meet the restrictions can always be broken up into several simpler questions. The program usually rejects questions it cannot handle, in which case the questioner may rephrase his question. He can also check the printed spec list to see if the computer is on the right track, in case the linguistic program has erred and failed to detect its own error. Finally, he can often judge whether the answer is reasonable.

Next Steps

No important difficulty is expected in augmenting the program to include logical connectives, negatives, and relation words. The inclusion of multiple-clause questions also seems fairly straightforward, if the questioner will mark off for the computer the boundaries of his clauses. The program can then deal with the subordinate clauses one at a time before it deals with the main clause, using existing routines. On the other hand, if the syntax analysis is required to determine the clause boundaries as well as the phrase structure, a much more sophisticated program would be required.

The problem of recognizing and resolving semantic ambiguities remains largely unsolved. Determining what is meant by the question "Did the Red Sox win most of their games in July?" depends on a much larger context than the immediate question. The computer might answer all meaningful versions of the question (we know of five), or might ask the questioner which meaning he intended. In general, the facility for the computer to query the questioner is likely to be the most powerful improvement. This would allow the computer to increase its vocabulary, to resolve ambiguities, and perhaps even to train the questioner in the use of the program.

Considerable pains were taken to keep the program general. Most of the program will remain unchanged and intact in a new context, such as voting records. The processing program will handle data in any sort of hierarchical form, and is indifferent to the attributes used. The syntax program is based entirely on parts of speech, which can easily be assigned to a new set of words for a new context. On the other hand, some of the subroutines contained in the dictionary meanings are certainly specific to baseball; probably each new context would require certain subroutines specific to it. Also, each context might introduce a number of modifiers and derived attributes that would have to be defined in terms of special subroutines for the processor. Hopefully, all such occasions for change have been isolated in a small area of special subroutines, so that the main routines can be unaltered. However, until we have actually switched contexts, we cannot say definitively that we have been successful in producing a general question-answering program.

Acknowledgment

The Baseball program was conceived by Fredrick C. Frick, Oliver G. Selfridge, and Gerald P. Dineen, whose continued guidance is

gratefully acknowledged.

References

1. A. Newell and F. Tonge, "An introduction to Information Processing Language V", Commun. Assoc. for Computing Mach., Vol. 3, pp. 205-211; April, 1960.

2. See the project summary, by Z. S. Harris, in Current Research and Development in Scientific Documentation No. 6, pp. 52-53, Nat'l Sciences Foundation, May, 1960.

3. L. Miller, J. Minker, W. G. Reed, and W. E. Shindle, "A multi-level file structure for information processing", Proceedings Western Joint Computer Conference, Vol. 17, pp. 53-59, May, 1960.

A BASIS FOR A MATHEMATICAL THEORY OF COMPUTATION, PRELIMINARY REPORT

John McCarthy
M.I.T. Computation Center
Cambridge, Massachusetts

Abstract: Programs that learn to modify their own behaviors require a way of representing algorithms so that interesting properties and interesting transformations of algorithms are simply represented. Theories of computability have been based on Turing machines, recursive functions of integers and computer programs. Each of these has artificialities which make it difficult to manipulate algorithms or to prove things about them. The present paper presents a formalism based on conditional forms and recursive functions whereby the functions computable in terms of certain base functions can be simply expressed. We also describe some of the formal properties of conditional forms, and a method called recursion induction for proving facts about algorithms.

Computation is sure to become one of the most important of the sciences. This is because it is the science of how machines can be made to carry out intellectual processes. We know that any intellectual process that can be carried out mechanically can be performed by a general purpose digital computer. Moreover, the limitations on what we have been able to make computers do so far seem to come far more from our weakness as programmers than from the intrinsic limitations of the machines. We hope that these limitations can be greatly reduced by developing a mathematical science of computation.

There are three established directions of mathematical research relevant to a science of computation. The first and oldest of these is numerical analysis. Unfortunately, its subject matter is too narrow to be of much help in forming a general theory, and it has only recently begun to be affected by the existence of automatic computation.

The second relevant direction of research is the theory of computability as a branch of recursive function theory. The results of the basic work in this theory including the existence of universal machines and the existence of unsolvable problems have established a framework in which any theory of computation must fit. Unfortunately, the general trend of research in this field has been to establish more and better unsolvability theorems, and there has been very little attention paid to positive results and none to establishing the properties of the kinds of algorithms that are actually used. Perhaps for this reason the formalisms for describing algorithms are too cumbersome to be used to describe actual algorithms.

The third direction of mathematical research is the theory of finite automata. Results which use the finiteness of the number of states tend not to be very useful in dealing with present computers which have so many states that it is impossible for them to go through a substantial fraction of them in a reasonable time.

The present paper is an attempt to create a basis for a mathematical theory of computation. Before mentioning what is in the paper, we shall discuss briefly what practical results can be hoped for from a suitable mathematical theory. This paper contains direct contributions towards only a few of the goals to be mentioned, but we list additional goals in order to encourage a gold rush.

1. To develop a universal programming language. We believe that this goal has been written off prematurely by a number of people. Our opinion of the present situation is that ALGOL is on the right track but mainly lacks the ability to describe different kinds of data, that COBOL is a step up a blind alley on account of its orientation towards English which is not well suited to the formal description of procedures, and that UNCOL is an exercise in group wishful thinking. The formalism for describing computations in this paper is not presented as a candidate for a universal programming language because it lacks a number of features, mainly syntactic, which are necessary for convenient use.

2. To define a theory of the equivalence of computation processes. With such a theory we can define equivalence preserving transformations. Such transformations can be used to take an algorithm from a form in which it is easily seen to give the right answers to an equivalent form guaranteed to give the same answers but which has other advantages such as speed, economy of storage, or the incorporation of auxiliary processes.

4. To represent algorithms by symbolic expressions in such a way that significant changes in the behavior represented by the algorithms are represented by simple changes in the symbolic expressions. Programs that are supposed to learn from experience change their behavior by changing the contents of the registers that represent the modifiable aspects of their behavior. From a certain point of view having a convenient representation of one's behavior available for modification is what is meant by consciousness.

5. To represent computers as well as computations in a formalism that permits a treatment of the relation between a computation and the computer that carries out the computation.

6. To give a quantitative theory of computation. There ought to be a quantitative measure of the size of a computation analogous to Shannon's measure of information. The present paper contains no information about this.

The present paper is divided into two sections. The first contains several descriptive formalisms with a few examples of their use, and the second contains what little theory we have that enables us to prove the equivalence of computations expressed in these formalisms. The formalisms treated are the following:

1. A way of describing the functions that are computable in terms of given base functions using conditional expressions and recursive function definitions. This formalism differs from those of recursive function theory in that it is not based on the integers or any other fixed domain.

2. Computable functionals, i.e. functions with functions as arguments.

3. Non-computable functions. By adjoining quantifiers to the computable function formalism, we obtain a wider class of functions which are not a priori computable. However, such functions can often be shown to be equivalent to computable functions. In fact, the mathematics of computation may have as one of its major aspects rules which permit us to transform functions from a non-computable form into a computable form.

4. Ambiguous functions. Functions whose values are incompletely specified may be useful in proving facts about functions where certain details are irrelevant to the statement being proved.

5. A way of defining new data spaces in terms of given base spaces and of defining functions on the new spaces in terms of functions on the base spaces. Lack of such a formalism is one of the main weaknesses of ALGOL, but the business data processing languages such as FLOWMATIC and COBOL have made a start in this direction, even though this start is hampered by concessions to the presumed prejudices of business men.

The second part of the paper contains a few mathematical results about the properties of the formalisms introduced in the first part. Specifically, we describe the following:

1. The formal properties of conditional expressions.

2. A method called recursion induction for proving the equivalence of recursively defined functions.

3. Some relations between the formalisms introduced in this paper and other formalisms current in recursive function theory and in programming.

We hope that the reader will not be angry about the contrast between the great expectations of a mathematical theory of computation and the meager results presented in this paper.

FORMALISMS FOR DESCRIBING COMPUTABLE FUNCTIONS AND RELATED ENTITIES

In this part we describe a number of new formalisms for expressing computable functions and related entities. The most important section is 1, the subject matter of which is fairly well understood. The other sections give formalisms which we hope will be useful in constructing computable functions and in proving theorems about them.

1. Functions Computable in Terms of Given Base Functions

Suppose we are given a base collection $\mathcal{F}$ of functions having certain domains and ranges. In the case of the non-negative integers, we may have the successor function and the predicate of equality, and in the case of the S-expressions discussed in (7), we have the five basic operations. Our object is to define a class of functions $C\{\mathcal{F}\}$ which we shall call the class of functions computable in terms of $\mathcal{F}$.

Before developing $C\{\mathcal{F}\}$ formally, we wish to give an example, and in order to give the example, we first need the concept of conditional expression. In our notation a conditional expression has the form

$$(p_1 \rightarrow e_1, p_2 \rightarrow e_2, \ldots p_n \rightarrow e_n)$$

which corresponds to the ALGOL 60 reference language (5) expression

if p_1 then e_1 else if p_2 then e_2...else if p_n then e_n

Here $p_1, \ldots, p_n$ are propositional expressions taking the values T of F standing for truth and falsity respectively.

The value of $(p_1 \rightarrow e_1, p_2 \rightarrow e_2, \ldots, p_n \rightarrow e_n)$ is the value of the e corresponding to the first p that has value T. Thus

$$(4 < 3 \rightarrow 7, 2 > 3 \rightarrow 8, 2 < 3 \rightarrow 9, 4 < 5 \rightarrow 7) = 9$$

Some examples of the conditional expressions for well known functions are

$$|x| = (x < 0 \rightarrow -x, x \not< 0 \rightarrow x)$$
$$\delta ij = (i = j \rightarrow 1, i \neq j \rightarrow 0)$$

and the triangular function whose graph is given in figure 1 is represented by the conditional expression

$$\mathrm{tri}(x) = (x < -1 \rightarrow 0, x \leq 0 \rightarrow x+1, x \leq 1 \rightarrow 1-x, x > 1 \rightarrow 0)$$

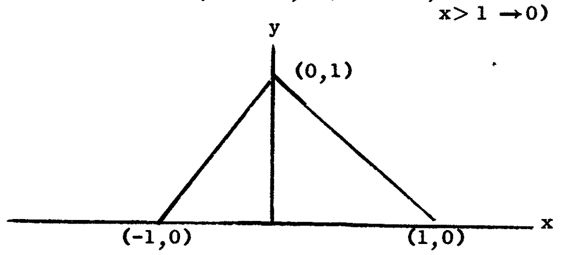


Figure 1

Now we are ready to use conditional expressions to define functions recursively. For example, we have

$$n = (n-0 \rightarrow 1, n \neq 0 \rightarrow n.(n-1)!)$$

Let us evaluate 2! according to this definition. We have

$$\begin{aligned} 2! &= (2=0 \rightarrow 1, 2\neq 0 \rightarrow 2\cdot(2-1)!) \\ &= 2\cdot 1 \\ &= 2(1=0 \rightarrow 1, 1\neq 0 \rightarrow 1\cdot(1-1)!) \\ &= 2\cdot 1\cdot 0! \\ &= 2\cdot 1\cdot(0=0 \rightarrow 1, 0\neq 0 \rightarrow 0\cdot(0-1)!) \\ &= 2\cdot 1\cdot 1 \\ &= 2 \end{aligned}$$

The reader who has followed these simple examples is ready for the construction of C $\{\mathcal{F}\}$ which is a straightforward generalization of the above together with a tying up of a few loose ends.

Some Notation. Let $\mathcal{F}$ be a collection (finite in the examples we shall give) of functions whose domains and ranges are certain sets. C $\{\mathcal{F}\}$ will be a class of functions involving the same sets which we shall call computable in terms of $\mathcal{F}$.

Suppose f is a function of n variables and suppose that if we write $y=f(x_1,\ldots,x_n)$, each x_i takes values in the set U_i and y takes its value in the set V. It is customary to describe this situation by writing

$$f:U_1 x U_2 x\ldots x U_n \rightarrow V$$

The set $U_1 x\ldots x U_n$ of n-tuplets $(x_1,\ldots,x_n)$ is called the domain of f, and the set V is called the range of f.

Forms and Functions. In order to make properly the definitions that follow, we will distinguish between functions and expressions involving variables. Following Church[1] the latter are called forms. Single letters such as f,g,h, etc. or sequences of letters such as sin are used to denote functions. Expressions such as f(x,y), f(g(x),y), x^2+y are called forms. In particular we may refer to the function f defined by $f(x,y)=x^2+y$. Our definitions will be written as though all forms involving functions were written f(,...,) although we will use expressions like x+y with infixes like ± in examples.

Composition. Now we shall describe the ways in which new functions are defined from old. The first way may be called (generalized) composition and involves the use of forms. We shall use the letters x,y,... (sometimes with subscripts) for variables and will suppose that there is a notation for constants that does not make expressions ambiguous. (Thus, the decimal notation is allowed for constants when we are dealing with integers.)

The class of forms is defined recursively as follows:

i) A variable x with an associated space U is a form, and with this form we also associate U. A constant in a space U is a form and we also associate U with this form.

ii) If $e_1,\ldots,e_n$ are forms associated with the spaces $U_1,\ldots,U_n$ respectively, then $f(e_1,\ldots,e_n)$ is a form associated with the space V. In this way the form (f(g(x,y),x) is built from the forms g(x,y) and x and the function f.

If all the variables occurring in a form e are among $x_1,\ldots x_n$ we can define a function h by writing $h(x_1,\ldots,x_n) = e$. We shall assume that the reader knows how to compute the values of a function defined in this way. If $f_1,\ldots,f_m$ are all the functions occurring in e we shall say that the function h is defined by composition from $f_1,\ldots,f_m$. The class of functions definable from given functions by composition only is narrower than the class of functions computable in terms of the given functions.

Partial Functions. In the theory of computation it is necessary to deal with partial functions which are not defined for all n-tuplets in their domains. Thus we have the partial function minus, defined by minus(x,y)=x-y, which is defined on those pairs (x,y) of positive integers for which x is greater than y. A function which is defined for all n-tuplets in its domain is called a total function. We admit the limiting case of a partial function which is not defined for any n-tuplets.

The n-tuplets for which a function described by composition is defined is determined in an obvious way from the sets of n-tuplets for which the functions entering the composition are defined. If all the functions occurring in a composition are total functions, the new function is also a total function, but the other processes for defining functions are not so kind to totality. When the work "function" is used from here on, we shall mean partial function.

Having to introduce partial functions is a nuisance, but an unavoidable one. The rules for defining computable functions sometimes give computation processes that never terminate, and when the computation process fails to terminate, the result is undefined. It has been shown that there is no effective general way of deciding whether a process will terminate.

Predicates and Propositional Forms

The space Π of truth values whose only elements are T (for truth) and F (for falsity) has a special role in our theory. A function whose range is Π is called a predicate. Examples of predicates on the integers are prime defined by

$$\text{prime}(x) = \begin{cases} T \text{ if } x \text{ is prime} \\ F \text{ otherwise} \end{cases}$$

and less defined by

$$\text{less}(x,y) = \begin{cases} T \text{ if } x<y \\ F \text{ otherwise} \end{cases}$$

We shall, of course, write $x<y$ instead of less(x,y). For any space U there is a predicate eq_U of two arguments defined by

$$eq_U(x,y) = \begin{cases} T \text{ if } x=y \\ F \text{ otherwise} \end{cases}$$

We shall write x=y instead of $eq_U(x,y)$, but some of the remarks about functions might not hold if we tried to consider equality a single predicate defined on all spaces at once.

A form with values in π such as $x<y$, $x=y$, or prime(x) is called a <u>propositional form</u>.

Propositional forms constructed directly from predicates such as prime(x) or $x<y$ may be called <u>simple</u>. Compound propositional forms can be constructed from the single ones by means of the propositional connectives $\wedge$, $\vee$, $\sim$. We shall assume that the reader is familiar with the use of these connectives.

<u>Conditional Forms or Conditional Expressions</u>. Conditional forms require a little more careful treatment than was given above in connection with the example. The value of the conditional form $(p_1 \to e_1, \ldots p_n \to e_n)$ is the value of the e corresponding to the first p that has value T; if all p's have value F, then the value of the conditional form is not defined. This rule is complete provided all the p's and e's have defined values, but we need to make provision for the possibility that some of the p's or e's are undefined. The rule is as follows

<u>If an undefined p occurs before a true p or if all p's are false or if the e corresponding to the first true p is undefined, then the form is undefined. Otherwise, the value of the form is the value of the e corresponding to the first true p.</u>

We shall illustrate this definition by additional examples:

$(2<1 \to 1, 2>1 \to 3) = 3$
$(1<2 \to 4, 1<2 \to 3) = 4$
$(2>1 \to 1, 3>1 \to 3)$ is undefined
$(0/0<1 \to 1, 1<2 \to 3)$ is undefined
$(1<2 \to 0/0, 1<2 \to 1)$ is undefined
$(1<2 \to 2, 1<3 \to 0/0) = 2$

The truth value T can be used to simplify certain conditional forms. Thus, instead of

$|x| = (x<0 \to -x, x \not< 0 \to x)$

we shall write

$|x| = (x<0 \to -x, T \to x)$

The propositional connectives can be defined by conditional forms as follows.

$p \wedge q = (p \to q, T \to F)$
$p \vee q = (p \to T, T \to q)$
$\sim p = (p \to F, T \to T)$
$p \subset q = (p \to q, T \to T)$

Considerations of truth tables shows that these formulas give the same results as the usual definitions. However, in order to treat partial functions we must consider the possibility that p or q may be undefined.

Suppose that p is false and q is undefined; then according to the conditional form definition $p \wedge q$ is false and $q \wedge p$ is undefined. This unsymmetry in the propositional connectives turns out to be appropriate in the theory of computation since if a calculation of p gives F as a result q need not be computed to evaluate $p \wedge q$, but if the calculation of p does not terminate, we never get around to computing p.

It is natural to ask if a function cond_n of 2n variables can be defined so that

$$(p_1 \to e_1, \ldots p_n \to e_n) = \text{cond}_n(p_1, \ldots, p_n, e_1, \ldots, e_n)$$

This is not possible unless we extend our notion of function because normally one requires all the arguments of a function to be given before the function is computed. However, as we shall shortly see, it is important that a conditional form be considered defined when, for example, p_1 is true and e_1 is defined and all the other p's and e's are undefined. The required extension of the concept of function would have the property that functions of several variables could no longer be identified with one-variable functions defined on product spaces. We shall not pursue this possibility further here.

We now want to extend our notion of forms to include conditional forms. Suppose $p_1, \ldots, p_n$ are forms associated with the space of truth values and $e_1, \ldots e_n$ are forms associated with the same space V. Suppose further that each variable x_i occurring in $p_1, \ldots p_n$ and $e_1, \ldots e_n$ is associated with the with the space U. Then $(p_1 \to e_1, \ldots, p_n \to e_n)$ is a form associated with V.

We believe that conditional forms will eventually come to be generally used in mathematics whenever functions are defined by considering cases. Their introduction is the same kind of innovation as vector notation. Nothing can be proved with them that could not also be proved without them. However, their formal properties, which will be discussed later, will reduce many case-analysis verbal arguments to calculation.

<u>Definition of Functions by Recursion</u>. The definition

$n! = (n=0 \to 1, T \to n \cdot (n-1)!)$

is an example of definition by recursion. Consider the computation of 0!

$0! = (0=0 \to 1, T \to 0 \cdot (0-1)!)$

We now see that it is important to provide that the conditional form is defined even if a term beyond the one that gives the value is undefined. In this case $(0-1)!$ is undefined.

Note also that if we consider a wider domain than the non-negative integers, n! as defined above becomes a partial function, since unless n is a non-negative integer, the recursion process does not terminate.

In general, we can either define single functions by recursion or define several functions together by simultaneous recursion, the former being a particular case of the latter.

To define simultaneously functions $f_1, \ldots f_k$, we write equations

$$f_1(x_1,\ldots,x_n) = e_1$$
$$\vdots$$
$$f_k(x_1,\ldots,x_n) = e_k$$

The expressions $e_1,\ldots,e_k$ must contain only known functions and the functions $f_1,\ldots,f_k$. Suppose that the ranges of the functions are to be $V_1,\ldots,V_k$ respectively; then we further require that the expressions $e_1,\ldots e_k$ be associated with these spaces respectively, given that within $e_1,\ldots e_k$ the f's are taken as having the V's as ranges. This is a consistency condition.

$f_i(x_i,\ldots x_k)$ is to be evaluated for given values of the x's as follows.

1. If e_i is a conditional form then the p's are to be evaluated in the prescribed order stopping when a true p and the corresponding e have been evaluated.
2. If e_i has the form $g(e_1^*,\ldots,e_m^*)$ then $e_1^*,\ldots,e_m^*$ are to be evaluated and then the func- g applied.
3. If any expression $f_i(e_1^*,\ldots e_{n_j}^*)$ occurs it is to be evaluated from the defining equation.
4. Any subexpressions of e_i that have to be evaluated are evaluated according to the same rules.
5. Variables occurring as subexpressions are evaluated by giving them the assigned values.

There is no guarantee that the evaluation process will terminate in any given case. If for particular arguments the process does not terminate, then the function is undefined for these arguments. The possibility of termination depends on the presence of conditional expressions in the e_i's.

The class of functions $C\{\mathcal{F}\}$ computable in terms of the given base functions $\mathcal{F}$ is defined to consist of the functions which can be defined by repeated applications of the above recursive definition process.

2. Recursive Functions of the Integers

In Reference 7 we develop the recursive functions of a class of symbolic expressions in terms of the conditional expression and recursive function formalism.

As an example of the use of recursive function definitions, we shall give recursive definitions of a number of functions over the integers. We do this for three reasons: to help the reader familiarize himself with recursive definition, to show how much simpler in practice our methods of recursive definition are than either Turing machines or Kleene's formalism, and to prove that any partial recursive function (Kleene) on the non-negative integers is in $C\{\mathcal{F}\}$ where $\mathcal{F}$ contains only the successor function and the predicate equality.

Let I be the set of non-negative integers $\{0,1,2,\ldots\}$ and denote the successor of an integer n by n' and denote the equality of integers n_1 and n_2 by $n_1=n_2$. If we define functions <u>succ</u> and <u>eq</u> by

$$\text{succ}(n) = n'$$
$$\text{eq}(n_1,n_2) = \begin{cases} T \text{ if } n_1 = n_2 \\ F \text{ if } n_1 \neq n_2 \end{cases}$$

then we write $\mathcal{F}_1 = \{\text{succ},\text{eq}\}$. We are interested in $C\{\mathcal{F}\}$. Clearly all functions in $C\{\mathcal{F}\}$ will have either integers or truth values as values.

First we define the predecessor function <u>pred</u> (not defined for n=0) by

$$\text{pred}(n) = \text{pred2}(n,0)$$
$$\text{pred2}(n,m) = (m'=n \rightarrow m. T \rightarrow \text{pred2}(n,m')).$$

We shall denote pred(n) by n^-.

Now we define the sum

$$m+n = (n=0 \rightarrow m, T \rightarrow m'+n^-),$$

the product

$$m\text{x}n = (n=0 \rightarrow 0, T \rightarrow m+m\text{x}n^-)$$

the difference

$$m-n = (n=0 \rightarrow m, T \rightarrow m^- - n^-)$$

which is defined only for m n, the inequality

$$m \leq n = (m=0) \vee (\sim(n=0) \wedge (m^- \leq n^-))$$

the strict inequality

$$m < n = (m \leq n) \wedge \sim(m=n)$$

the integer valued quotient

$$m/n = (m < n \rightarrow 0, T \rightarrow ((m-n)/n)'),$$

the remainder

$$\text{rem}(m/n) = (m < n \rightarrow m, T \rightarrow \text{rem}(m-n/n))$$

and the divisibility of a number n by a number m

$$m|n = (n=0) \vee ((n \geq m) \wedge (m|(n-m))).$$

The primeness of a number is defined by

$$\text{prime}(n) = (n\neq 0) \wedge (n\neq 1) \wedge \text{prime } 2(n,2)$$

where

$$\text{prime2}(m,n) = (m=n) \vee ((m \nmid n) \wedge \text{prime2}(n,m'))$$

The Euclidean algorithm defines the greatest common divisor if we write

$$\text{gcd}(m,n) = (m > n \rightarrow \text{gcd}(n,m), \text{rem}(n/m) = 0 \rightarrow m, T\ \text{gcd}(\text{rem}(m/m),m))$$

and we can define Euler's φ-function by

$$\varphi(n) = \varphi 2(n,n)$$

where

$$\varphi 2(n,m) = (m=1 \rightarrow 1,\ \text{gcd}(n,m) = 1 \rightarrow \varphi 2(n,m^-)',\ T \rightarrow \varphi 2(n,m^-))$$

The above shows that our form of recursion is a convenient way of defining arithmetical functions. We shall see how the properties of the arithmetical functions can conveniently be derived in this formalism in a later section.

3. Computable Functionals

The formalism previously described enables us to define functions that have functions as arguments. For example,

$$\sum_{i=m}^{n} a_i$$

can be regarded as a function of the numbers <u>m</u> and <u>n</u> and the sequence $\{a_i\}$. If we regard the sequence as a function f we can write the recursive definition

$$\text{sum}(m,n,f) = (m > n \rightarrow 0, T \rightarrow f(m) + \text{sum}(m+1,n,f))$$

or in terms of the conventional notation

$$\sum_{i=m}^{n} f(i) = (m > n \rightarrow 0, T \rightarrow f(m) + \sum_{i=m+1}^{n} f(i))$$

Functions with functions as arguments are called functionals.

Another example is the functional least(p) which gives the least integer n such that p(n) for a predicate p. We have

$$\text{least}(p) = \text{least2}(p,0)$$

where

$$\text{least2}(p,n) = (p(n) \rightarrow n, T \rightarrow \text{least2}(p,n+1))$$

In order to use functionals it is necessary to have a notation for naming functions. We use Church's[1] lambda notation. Suppose we have a function f defined by an equation $f(x_1,\ldots,x_n)=e$ where e is some expression in $x_1,\ldots,x_n$.

The name of this function is $\lambda((x_1,\ldots x_n),e)$. For example, the name of the function f defined by $f(x,y) = x^2+y$ is $\lambda((x,y),x^2+y)$. We have $\lambda((x,y),x^2+y)(3,4) = 13$ but $\lambda((y,x),x^2+y)(3,4) = 19$.

The variables occurring in a λ definition are dummy or bound variables and can be replaced by others without changing the function provided the replacement is done consistently. For example, the expressions $\lambda((x,y),x^2+y)$ and $\lambda((u,v),u^2+v)$ and $\lambda((y,x),y^2+x)$ all represent the same function.

In the notation $\sum_{i=1}^{n} i^2$ is represented by $\text{sum}(1,n,\lambda((i),i^2))$ and the least integer n for which $n^2 > 50$ is represented by

$$\text{least}(\lambda((n),n^2 > 50)$$

When the functions with which we are dealing are defined recursively, a difficulty arises. For example, consider factorial defined by

$$\text{factorial}(n) = (n=0 \rightarrow 1, T \rightarrow n\cdot\text{factorial}(n-1))$$

The expression

$$\lambda((n),(n=0 \rightarrow 1, T \rightarrow n\cdot\text{factorial}(n-1)))$$

cannot serve as a name for this function because it is not clear that the occurrence of "factorial" in the expression refers to the function defined by the expression as a whole. Therefore, for recursive functions we adopt an additional convention. Namely,

$$\text{label}(f,\lambda((x_1,\ldots x_n),e))$$

stands for the function f defined by the equation

$$f(x_1,\ldots x_n) = e$$

where any occurrences of the function letter f within e stand for the function being defined. The letter f also serves as a dummy variable. The factorial function then has the name

$$\text{label}(\text{factorial}, \lambda((n),(n=0 \rightarrow 1, T \rightarrow n\cdot\text{factorial}(n-1))))$$

and since factorial and n are dummy variables the expression

$$\text{label}(g,\lambda((r),(r=0 \rightarrow 1, T \rightarrow r\cdot g(r-1))))$$

represents the same function.

If we start with a base domain for our variables, it is possible to consider a hierarchy of functionals. At level 1 we have functions whose arguments are in the base domain. At level 2 we have functionals taking functions of level 1 as arguments. At level 3 are functionals taking functionals of level 2 as arguments etc. Actually functionals of several variables can be of mixed type.

However, this hierarchy does not exhaust the possibilities, and if we allow functions which can take themselves as arguments we can eliminate the use of label in naming recursive functions. Suppose that we have a function f defined by

$$f(x) = \mathscr{E}(x,f)$$

where $\mathscr{E}(x,f)$ is some expression in x and the function variable f. This function can be named

$$\text{label}(f,\lambda((x),\mathscr{E}(x,f)))$$

However, suppose we define a function g by

$$g(x,\varphi) = \mathscr{E}(x,\lambda((x),\varphi(x,\varphi)))$$

or

$$g = \lambda((x,\varphi),\mathscr{E}(x,\lambda((x),\varphi(x,\varphi))))$$

We then have

$$f(x) = g(x,g)$$

since g(x,g) satisfies the equation

$$g(x,g) = \mathscr{E}(x,\lambda((x),g(x,g)))$$

Now we can write f as

$$f = \lambda((x),\lambda((y,\varphi),\mathscr{E}(y,\lambda((u),\varphi(u,\varphi)))) (x,\lambda((y,\varphi)\mathscr{E}(y,\lambda((u),\varphi(u,\varphi))))$$

This eliminates label at what seems to be an excessive cost. Namely, the expression gets quite complicated and we must admit functionals capable of taking themselves as arguments thus escaping our orderly hierarchy of functionals.

4. Non-Computable Functions and Functionals

It might be supposed that in a mathematical theory of computation one need only consider computable functions. However, mathematical physics is carried out in terms of real valued functions which are not computable but only approximable by computable functions.

We shall consider several successive extensions of the class $C\{\mathscr{F}\}$. First we adjoin the universal quantifier $\forall$ to the operations used to define new functions. Suppose e is a form in a variable x and other variables associated with the space π of truth values. Then

$$\forall((x),e)$$

is a new form in the remaining variables also associated with π. $\forall((x),e)$ has the value T for given values of the remaining variables if for all values of x, e has the value T. $\forall((x),e)$ has the value F if for at least one value of x, e has the value F. In the remaining case, i.e. for some values of x e has the value T and for all others e is undefined, $\forall((x),e)$ is undefined.

If we allow the use of the universal quantifier to form new propositional forms for use in conditional forms, we get a class of functions $Ha\{\mathscr{F}\}$ which may well be called the class of functions hyper-arithmetic over $\mathscr{F}$ since in the case where $\mathscr{F} = \{\text{successor, equality}\}$ on the integers, $Ha\{\mathscr{F}\}$ consists of Kleene's[5] hyper-arithmetic functions.

Our next step is to allow the description operator ι. $\iota((x),\pi(x))$ stands for the unique

$\underline{x}$ such that $\underline{\pi(x)}$ is true. Unless there is such an $\underline{x}$ and it is unique $\iota((x),\pi(x))$ is undefined. In the case of the integers $\iota((x),\pi(x))$ can be defined in terms of the universal quantifier using conditional expressions, but this does not seem to be the case in domains which are not effectively enumerable, and one may not wish to do so in domains where enumeration is unnatural.

The next step is to allow quantification over functions. This gets us to Kleene's[5] analytic hierarchy and presumably allows the functions used in analysis. Two facts are worth noting. First $\forall((f),\varphi(f))$ refers to all functions on the domain and not just the computable ones. If we restrict quantification to computable functions, we get different results. Secondly, if we allow functions which can take themselves as arguments, it is difficult to assign a meaning to the quantification. In fact, we are apparently confronted with the paradoxes of naive set theory.

5. Ambiguous Functions

Ambiguous functions are not really functions. For each prescription of values to the arguments the ambiguous function has a collection of possible values. An example of an ambiguous function is $\underline{\text{less}(n)}$ defined for all positive integer values of $\underline{n}$. Every non-negative integer less than $\underline{n}$ is a possible value of $\underline{\text{less}(n)}$. If we define a basic ambiguity operator $\underline{\text{amb}(x,y)}$ whose possible values are $\underline{x}$ and $\underline{y}$ when both are defined otherwise whichever is defined, we can define $\underline{\text{less}(n)}$ by

$$\text{less}(n) = \text{amb}(n-1,\text{less}(n-1)).$$

less(n) has the property that if we define

$$\text{ult}(n) = (n=0 \rightarrow 0, T \rightarrow \text{ult}(\text{less}(n)))$$

then

$$\forall((n),\text{ult}(n)=0) = T.$$

There are a number of important kinds of mathematical arguments whose convenient formalization may involve ambiguous functions. In order to give an example, we need two definitions. If $\underline{f}$ and $\underline{g}$ are two ambiguous functions, we shall say that $\underline{f}$ is a descendant of $\underline{g}$ if for each $\underline{x}$ every possible value of $\underline{f(x)}$ is also a possible value of $\underline{g(x)}$. Secondly, we shall say that a property of ambiguous functions is hereditary if whenever it is possessed by a function $\underline{g}$ it is also possessed by all descendants of $\underline{g}$. The property that iteration of an integer valued function eventually gives 0 is hereditary and the function $\underline{\text{less}}$ has this property. So, therefore, do all its descendants. Thus any function, however, complicated which always reduces a number will if iterated sufficiently always give 0.

This example is one of our reasons for hoping that ambiguous functions will turn out to be useful.

With just the operation amb defined above adjoined to those used to generate $C\{\mathcal{F}\}$, we can extend $\mathcal{F}$ to the class $C^*\{\mathcal{F}\}$ which may be called the computably ambiguous functions. A wider class of ambiguous functions if formed using the operator $\text{Am}(x,\pi(x))$ whose values are all $\underline{x\text{'s}}$ satisfying $\pi(x)$.

6. Recursive Definitions of Sets

In the previous sections on recursive definition of functions the domains and ranges of the basic functions were prescribed and the defined functions had the same domains and ranges.

In this section we shall consider the definition of new sets and the basic functions on them. First we shall consider some operations whereby new sets can be defined.

1. The Cartesian product AxB of two sets A and B is the set of all ordered pairs (a.b) with $a \in A$ and $b \in B$. If A and B are finite sets and n(A) and n(B) denote the numbers of members of A and B respectively then $n(A\text{x}B=n(A)\cdot n(B)$.

Associated with the pair of sets (A,B) are two canonical mappings

$\pi_{A,B}: A\text{x}B \rightarrow A$ defined by $\pi_{A,B}((a\cdot b)) = a$

$\rho_{A,B}: A\text{x}B \rightarrow B$ defined by $\rho_{A,B}((a\cdot b)) = b$

The word "canonical" refers to the fact that $\pi_{A,B}$ and $\rho_{A,B}$ are defined by the sets A and B and do not depend on knowing anything about the members of A and B.

The next canonical function γ is a function of two variables $\gamma_{A,B}: A,B \rightarrow A\text{x}B$ defined by

$\gamma_{A,B}(a,b) = (a\cdot b)$

For some purposes functions of two variables x from A and y from B can be identified with functions of one variable defined on AxB.

2. The direct union $A\oplus B$ of the sets A and B is the union of two non-intersecting subsets one of which is in 1-1 correspondence with A and the other with B. If A and B are finite, then $n(A\oplus B) = n(A)+n(B)$ even if A and B intersect. The elements of $A\oplus B$ may be written as elements of A or B subscripted with the set from which they come, i.e. a_A or b_B.

The canonical mappings associated with the direct union $A\oplus B$ are

$i_{A,B}: A \rightarrow A\oplus B$ defined by $i_{A,B}(a) = a_A$

$j_{A,B}: B \rightarrow A\oplus B$ defined by $j_{A,B}(b) = b_B$

$p_{A,B}: A\oplus B \rightarrow \pi$ defined by $p_{A,B}(x) = T$ if and only if x comes from A.

$q_{A,B}: A\oplus B \rightarrow \pi$ defined by $q_{A,B}(x) = T$ if and only if x comes from B

There are also two canonical partial functions $r_{A,B}: A\oplus B \rightarrow A$ which is defined only for elements coming from A and satisfies $r_{A,B}(\ _{A,B}(a))=a$. Similarly $s_{A,B} A\oplus B \rightarrow B$ satisfies $s_{A,B}(j_{A,B}(b))=b$.

3. The power set A^{B} is the set of all mappings $f: B \rightarrow A$. The canonical mapping $\varepsilon_{A,B}: A^{B}\text{x}B \rightarrow A$ is defined by $\varepsilon_{A,B}(f,b) = f(b)$

Canonical Mappings

We will not regard the sets Ax(BxC) and (AxB)xC as the same, but there is a canonical 1-1 mapping between them

$g_{A,B,C}: (A\text{x}B)\text{x}C \rightarrow A\text{x}(B\text{x}C)$

defined by

$$g_{A,B,C}(u) = \gamma_{A,B\times C}(\pi_{A,B}(\pi_{A\times B,C}(u)), \gamma_{B,C}(\rho_{A,B}(\pi_{A\times B,C}(u)), \rho_{A\times B,C}(u))).$$

We shall write

$$(A\times B)\times C \cong A\times(B\times C)$$

to express the fact that these sets are canonically isomorphic.

Other canonical isomorphisms are

1. $t_{A,B}: A\times B \rightarrow B\times A$ defined by
   $t(u) = \gamma_{B,A}(\rho_{A,B}(u), \pi_{A,B}(u))$
2. $d_1: A\times(B\oplus C) \rightarrow A\times B\oplus A\times C$
3. $a_2: (A\oplus B)\oplus C \rightarrow A\oplus(B\oplus C)$
4. $d_2: A^C \times B^C \rightarrow (A\times B)^C$
5. $d_3: A^B \times A^C \rightarrow A^{B\oplus C}$
6. $s_1: (A^B)^C \rightarrow A^{B\times C}$

We shall denote the null set (containing no elements) by 0 and the set consisting of the integers from 1 to n by n. We have

$A\oplus 0 \cong A$
$A\oplus 0 \cong 0$
$A\times 1 \cong A$
$A\times 2 \cong A\oplus A$ (n terms, associate to left by convention)
$A^0 \cong 1$ (by convention)
$A^1 \cong A$
$A \cong A\times\ldots\times A$ (n terms, associate to left by convention)

Suppose we write the recursive equation

$S = \{\Lambda\}\oplus A\times S$

we can interpret this as defining the set of sequences of elements of A as follows:

1. Interpret Λ as denoting the null sequence, Then the null sequence (strictly an image of it) is an element of S.

2. Since a pair consisting of an element of A and an element of S is an element of S, a pair (a,Λ) is an element of S. So, then, are

$a_1\cdot(a_2\cdot\Lambda))$ and $a_1\cdot(a_2\cdot(a_3\cdot\Lambda)))$ etc.

Thus S consists of all sequences of elements of A including the null sequence.

Suppose we substitute $\{\Lambda\}\oplus A\times S$ for S in the right side of $S = \{\Lambda\}\oplus A\times S$. We get

$S = \{\Lambda\}\oplus A\times(\{\Lambda\}\oplus A\times S)$

If we again substitute for S and expand by the distributive law we get

$S = \{\Lambda\}\oplus A\times\{\Lambda\}\oplus A\times A\times\{\Lambda\}+\ldots.$

which if we denote the set $\{\Lambda\}$ becomes

$S = 1\oplus A\oplus A^2\oplus A^3\oplus\ldots$

which is another way of writing the set of sequences. We shall denote the set of sequences of elements of A by <u>seq</u>(A).

Another useful recursive construction is

$S = A\oplus S\times S$

Its elements have the forms a or $(a_1\cdot a_2)$ or $((a_1\cdot a_2)\cdot a_3)$ or $(a_1\cdot(a_2\cdot a_3))$ etc. Thus we have the set of S-expressions on the alphabet A which we may denote by sexp(A). This set is the subject matter of Reference 7 and the following paragraph refers to this paper.

When sets are formed by this kind of recursive definition, the canonical mappings associated with the direct sum and Cartesian product operations have significance. Consider, for example, <u>sexp</u>(A).

We can define the basic operations of LISP, i.e. atom, eq, car, cdr and cons by the equations

$\text{atom}(x) = p_{A,S\times S}(x)$

$\text{eq}(x,y) = (i_{A,S\times S}(x) = i_{A,S\times S}(y))$

assuming that equality is defined on the space A

$\text{car}(x) = \pi_{S,S}(\sigma_{A,S\times S}(x))$

$\text{cdr}(x) = \rho_{S,S}(\sigma_{A,S\times S}(x))$

$\text{cons}(x,y) = j_{A,S\times S}(\gamma_{S,S}(x,y))$

<u>Definition of the set of integers.</u> Let Λ denote the null set and $\{\Lambda\}$ be the set whose sole element is the null set. We can define the set of integers I by

$I = \{\Lambda\}\oplus\{\Lambda\}\times I$

its elements are then

$\Lambda, (\Lambda,\Lambda), (\Lambda,(\Lambda,\Lambda)), (\Lambda,(\Lambda,(\Lambda,\Lambda)))$ etc.

which we shall denote by 0,1,2,3 etc.
The successor and predecessor functions are then definable in terms of the canonical operations of the defining equation. We have

$\text{succ}(n) = \gamma_{\{\Lambda\},I}(\Lambda,n)$

$\text{pred}(n) = \rho_{\{\Lambda\},I}(\sigma_{\{\Lambda\},\{\Lambda\}\times I}(n))$

PROPERTIES OF COMPUTABLE FUNCTIONS

The first part of this paper was solely concerned with presenting descriptive formalisms, In this part we shall establish a few of the properties of the entities we previously introduced. The most important section is the second dealing with recursion induction.

7. <u>Formal Properties of Conditional Forms</u>

The theory of conditional expressions corresponds to analysis by cases in mathematics and is only a mild generalization of propositional calculus.

We start by considering expressions called generalized Boolean forms (gbf) formed as follows:

1. Variables are divided into propositional variables p,q,r, etc. and general variables x,y,z, etc.

2. We shall write $(p \rightarrow x,y)$ for $(p \rightarrow x, T \rightarrow y)$. $(p \rightarrow x,y)$ is called an elementary conditional form (ecf) of which p,x, and y are called the <u>premise</u>, <u>conclusion</u> and the <u>alternative</u>, respectively.

3. A variable is a gbf and if it is a propositional variable it is called a pf (propositional form).

4. If π is a <u>pf</u> and α and β are gbfs, then $(\pi \rightarrow \alpha, \beta)$ is a gbf. If, in addition, α and β are pfs so is $(\pi \rightarrow \alpha, \beta)$.

The value of a <u>gbf</u> α for given values (T,F or undefined) of the propositional variables will be T or F in case α is a <u>pf</u> or a general variable otherwise. This value is determined for a <u>gbf</u> $(\pi \rightarrow \alpha, \beta)$ according to the table

value (π)	value $((\pi \rightarrow \alpha, \beta))$
T	value (α)
F	value (β)
undefined	undefined

We shall say that two gbfs are strongly equivalent if they have the same value for all values of the propositional variables in them including the case of undefined propositional variables. They are weakly equivalent if they have the same values for all values of the propositional variables when these are restricted to T and F.

The equivalence of gbfs can be tested by a method of truth tables identical to that of propositional calculus. The table for $((p \rightarrow q,r) \rightarrow a,b)$ and $(p \rightarrow (q \rightarrow a,b)(r \rightarrow a,b))$ is

pqr	$(p \rightarrow q,r)$	$((p \rightarrow q,r) \rightarrow a,b)$	$(q \rightarrow a,b)$	$(r \rightarrow a,b)$	$(p \rightarrow (q \rightarrow a,b),(r \rightarrow a,b))$
TTT	T	a	a	a	a
TTF	T	a	a	b	a
TTu	T	a	a	u	a
TFT	F	b	b	a	b
TFF	F	b	b	b	b
TFu	F	b	b	u	b
TuT	u	u	u	a	u
TuF	u	u	u	b	u
Tuu	u	u	u	u	u
FTT	T	a	a	a	a
FTF	F	b	a	b	b
FTu	u	u	a	u	u
FFT	T	a	b	a	a
FFF	F	b	b	b	b
FFu	u	u	b	u	u
FuT	T	a	u	a	a
FuF	F	b	u	b	b
Fuu	u	u	u	u	u
uTT	u	u	a	a	u
uTF	u	u	a	b	u
uTu	u	u	a	u	u
uFT	u	u	b	a	u
uFF	u	u	b	a	u
uFu	u	u	b	u	u
uuT	u	u	u	a	u
uuF	u	u	u	b	u
uuu	u	u	u	u	u

According to the table $((p \rightarrow q,r) \rightarrow a,b)$ and $(p \rightarrow (q \rightarrow a,b),(r \rightarrow a,b))$ are strongly equivalent.

For weak equivalence the u case can be left out of the table. Consider the table

p q	$(q \rightarrow a,b)$	$(q \rightarrow c,d)$	$(p \rightarrow (q \rightarrow a,b),(q \rightarrow c,d))$	$(p \rightarrow a,c)$	$(p \rightarrow b,d)$	$(q \rightarrow (p \rightarrow a,c),(p \rightarrow b,d))$
T T	a	c	a	a	b	a
T F	b	d	b	a	b	b
F T	a	c	c	c	d	c
T T	b	d	d	c	d	d

which proves that $(p \rightarrow (q \rightarrow a,b),(q \rightarrow c,d))$ and $(q \rightarrow (p \rightarrow a,c),(p \rightarrow b,d))$ are weakly equivalent. They are also strongly equivalent. We shall write $\equiv_s$ and $\equiv_w$ for the relations of strong and weak equivalence.

There are two rules whereby an equivalence can be used to generate other equivalences.

1. If $\alpha \equiv \beta$ and $\alpha_1 \equiv \beta_1$ is the result of substituting any gbf for any variable in $\alpha \equiv \beta$, then $\alpha_1 \equiv \beta_1$. This is called the rule of substitution.

2. If $\alpha \equiv \beta$ and α is sub-expression of γ and δ is the result of replacing an occurrence of α in γ by an occurrence of β, then $\gamma \equiv \delta$ This is called the rule of replacement.

These rules are applicable to either strong or weak equivalence and in fact to much more general situations.

Weak equivalence corresponds more closely to equivalence of truth functions in propositional calculus than does strong equivalence.

Consider the equations

1) $(p \rightarrow a,a) \equiv_w a$

2) $(T \rightarrow a,b) \equiv_s a$

3) $(F \rightarrow a,b) \equiv_s b$

4) $(p \rightarrow T,F) \equiv_s p$

5) $(p \rightarrow (p \rightarrow a,b),c) \equiv_s (p \rightarrow a,c)$

6) $(p \rightarrow a,(p \rightarrow b,c)) \equiv_s (p \rightarrow a,c)$

7) $((p \rightarrow q,r) \rightarrow a,b) \equiv_s (p \rightarrow (q \rightarrow a,b), (r \rightarrow a,b))$

8) $((p \rightarrow (q \rightarrow a,b),(q \rightarrow c,d)) \equiv_s (q \rightarrow (p \rightarrow a,c),(p \rightarrow b,d))$

All are strong equivalence except the first and can be proved by truth tables.

These eight equations can be used as axioms to transform an gbf into any weakly equivalent one using substitution and replacement. In fact, they can be used to transform any gbf into a canonical form. This canonical form is the following. Let $p_1,\dots,p_n$ be the variables of the gbf a taken in an arbitrary order. Then a can be transformed into the form

$(p_1 \rightarrow a_0,a_1)$

where each a_1 has the form

$a_1 = (p_2 \rightarrow a_{10},a_{11})$

and in general for each k = 1,n-1

$$a_{i_1},\ldots,_{i_k} = (p_{i+1} \rightarrow a_{i_1},\ldots,_{i_k},0, a_{i_1}\ldots_{i_k},1)$$

and each $a_{i_1},\ldots,_{i_n}$ is a truth value or a general variable.

For example, the canonical form of

$$((p \rightarrow q,r) \rightarrow a,b)$$

with the variables taken in the order r,q,p is

$$(r \rightarrow (q \rightarrow (p \rightarrow a,a),(p \rightarrow b,a)),(q \rightarrow (p \rightarrow a,b),(p \rightarrow b,b)))$$

In this canonical form, the 2^n cases of the truth or falsity of $p_1,\ldots,p_n$ are explicitly exhibited.

An expression may be transformed into canonical form as follows:

1) Axiom 7 is used repeatedly until in every sub-expression the π in $(\pi \rightarrow \alpha_1, \beta)$ consists of a single propositional variable.

2) The variable p_1 is moved to the front by repeated application of axiom 8. There are three cases: $(q \rightarrow (p_1 \rightarrow a,b),(p_1 \rightarrow c,d))$ to which axiom 8 is directly applicable.
$(q \rightarrow a,(p_1 \rightarrow c,d))$ where axiom 8 becomes applicable after axiom 1 is used to make it
$(q \rightarrow (p_1 \rightarrow a,a),(p_1 \rightarrow c,d))$ and then axiom 8 is applied, the case $(q \rightarrow (p_1 \rightarrow a,b),c)$ is handled similarly.

Once the main expression has the form $(p_1 \rightarrow \alpha, \beta)$ we move any p_1's which occur in α and β to the front and eliminate them using axioms 5 and 6. We then bring p_2 to the front of α and β using axiom 1 if necessary to guarantee at least one occurrence of p_2 in each of α and β. The process is continued until the canonical form is achieved.

There is also a canonical form for strong equivalence. Any gbf a is strongly equivalent to one of the form $(p_1 \rightarrow \alpha, \beta)$, where α and β do not contain p_1 and are themselves in canonical form. However, the variable p_1 may not be chosen arbitrarily but must be an inevitable propositional variable of the original gbf and can be chosen to be any inevitable variable. An inevitable variable of a gbf $(\pi \rightarrow \alpha, \beta)$ is defined to be either the first propositional variable or else an inevitable variable of both α and β.

A gbf a may be put in strong canonical form as follows:

1) Use axiom 7 to get all premises as propositional variables.

2) Choose any inevitable variable, say p_1, and put a in the form $(p_1 \rightarrow \alpha, \beta)$ by using axiom 8.

3) The next step is to eliminate occurrences of p_1 in α and β. This can be done by the general rule that in any ecf occurrences of the premise in the conclusion can be replaced by T and occurrences in the alternative by F. However, if we wish to use substitution and replacement on formulas we need the additional axioms

(9) $(p \rightarrow (q \rightarrow a,b),c) \equiv_s (p \rightarrow (q \rightarrow (p \rightarrow a,a),(p \rightarrow b,b)),c)$

and

(10) $(p \rightarrow a,(q \rightarrow b,c)) \equiv_s (p \rightarrow a,(q \rightarrow (p \rightarrow b,b),(p \rightarrow c,c)))$

Suppose there is an occurrence of p_1 in the conclusion; we want to replace it by T. To do this, we use axioms 9 and 10 to move in a p_1 until the objectionable p_1 occurs as the inner p_1 of one of the forms

$$(p_1 \rightarrow (p_1 \rightarrow a,b),c)$$

or

$$(p_1 \rightarrow a,(p_1 \rightarrow b,c)).$$

In either case, the objectionable p_1 can be removed by axiom 5 or 6 and the p_1's that were moved in can be moved out again.

Thus we have $(p_1 \rightarrow \alpha, \beta)$ with p_1 missing from α and β.

4) Inevitable variables are then brought to the front of α and β and so forth.

Two gbfs are equivalent (weakly or strongly) if and only if they have the same(weak or strong) canonical form. One way this is easy to prove; if two gbfs have the same canonical form they can be transformed into each other via the canonical form. Suppose two gbfs have different weak canonical forms when the variables are taken in the same order. Then values can be chosen for the p's giving different values for the form proving non-equivalence. In the strong case, suppose that two gbfs do not have the same inevitable propositional variables. Let p be inevitable in a but not in b. Then if the other variables are assigned suitable values b will defined with p undefined. However, a will be undefined since p is inevitable in a which proves non-equivalence. Therefore, strongly equivalent gbfs have the same inevitable variables, so let one of them be put in front of both gbfs. The process is then repeated in the conclusion and alternative etc.

The general kind of conditional expression

$$(p_1 \rightarrow e_1,\ldots,p_n \rightarrow e_n)$$

can be regarded as having the form

$$(p_1 \rightarrow e_1,(p_2 \rightarrow e_2,\ldots(p_n \rightarrow e_n,u)\ldots))$$

where u is a special undefined variable and their properties can be derived from those of ecc's.

The relation of functions to conditional expressions is given by the distributive law

$$f(x_1,\ldots,x_{i-1},(p_1 \rightarrow e_1,\ldots,p_n \rightarrow e_n), x_{i+1},\ldots x_k) =$$
$$(p_1 \rightarrow f(x_1,\ldots x_{i-1},e_1,x_{i+1},\ldots,x_n),\ldots,p_n \rightarrow f(x_1,\ldots,x_{i+1},e_n,x_{i+1},\ldots,x_n))$$

The rule of replacement can be extended in the case of conditional expressions. Suppose α is an occurrence of a sub-expression of an expression . We define a certain propositional expression π called the premise of α in β as follows:

1) The premise of α in α is T.

2) The premise of β in $f(x_1,\ldots,\ _n,\ldots x_n)$ where β is part of α is the premise of α in β .

3) The premise of β in $(p_1 \rightarrow e_1,\ldots p_i \rightarrow e_i,\ldots p_n \rightarrow e_n)$ where β occurs in e_i, and the premise of α in e_i is π, is $\sim p_1 \wedge \ldots \wedge \sim p_{i-1} \wedge p_i \wedge \pi$.

4) The premise of β in $(p_1 \rightarrow e_1,\ldots p_i \rightarrow e_i,\ldots p_n \rightarrow e_n)$ where β occurs in p_i, and the premise of α in p_i is π, is $\sim p_1 \wedge \ldots \wedge \sim p_{i-1} \wedge \pi$.

The extension of the rule of replacement is that an occurrence of α in β may be replaced by α' if $(\pi \rightarrow \alpha) \equiv_s (\pi \rightarrow \alpha')$ where π is the premise of α in β . Thus in a subcase one need only prove equivalence under the premise of the subcase.

8. Recursion Induction

Suppose a function f is defined recursively by

1) $f(x_1,\ldots,x_n) = \mathcal{E}\{x_1,\ldots x_n, f\}$

where $\mathcal{E}$ is an expression that in general contains f. Suppose that $\mathcal{A}$ is the set of n-tuplets $(x_1,\ldots,x_n)$ for which f is defined. Now let g and h be two other functions with the same domain as f and which are defined for all n-tuplets in $\mathcal{A}$. Suppose further that g and h satisfy the equation which defined f. We assert that

$g(x_1,\ldots,x_n) = h(x_1,\ldots x_n)$

for all $(x_i,\ldots,x_n)$ in $\mathcal{A}$. This is so simply because equation 1) uniquely determines the value that any function satisfying it has for arguments in $\mathcal{A}$ which in turn follows from the fact that 1) can be used to compute $f(x_1,\ldots,x_n)$ for $(x_1,\ldots x_n)$ in $\mathcal{A}$.

We shall call this method of proving two functions equivalent by the name of recursion induction.

We shall develop some of the properties of the elementary functions of integers in order to illustrate proof by recursion induction. We recall the definitions

$m+n = (n=0 \rightarrow m, T \rightarrow m'+n^-)$

$mn = (n=0 \rightarrow 0, T \rightarrow m+mn^-)$

Th 1. $m+0 = m$

Proof $m+0 = (0=0 \rightarrow m, T \rightarrow m'+0^-)$
$= m$

Only the definition of addition and the properties of conditional expressions were used in this proof

Th 2 $(m+n)' = m'+n$

Proof Define $f(m,n) = (n=0 \rightarrow m', T \rightarrow f(m,n^-))$

It is easily seen that f(m,n) converges for all m and n and hence is defined by the equation.

$(m+n)' = (n=0 \rightarrow m, T \rightarrow m'+n^-)'$
$= (n=0 \rightarrow m', T \rightarrow (m'+n^-)')$
$m+n = (n=0 \rightarrow m', T \rightarrow (m')+n^-)$

It is easily seen that the functions g and h defined by the equations $g(m,n) = (m+n)'$ and $h(m,n) = m'+n$ both satisfy the equation f. For example, it is clear that $g(m',n^-) = (m'+n^-)'$ and $h(m',n^-) = (m')'+n^-$. Therefore, by the principle of recursion induction h and g are equivalent functions on the domain of where f is defined, but this is the set of all pairs of integers.

The fact that f(m,n) converges for all m and n is a case of the fact that all functions defined by equations of the form

$f(n,x_1,\ldots x_n) = (n=0 \rightarrow g(x_1\ldots x_n), T \rightarrow h(f(n^-, r_1'(x_1,\ldots x_n),\ldots r_n'(x_1\ldots x_n)),\ldots, f(n^-, r_1^k(x_1,\ldots,x_n),\ldots, r_n^k(x_1\ldots x_n), n, x_1\ldots x_n))$

converge. We shall postpone discussing formal proofs of convergence.

In presenting further proofs we shall be more terse.

Th 3. $(m+n)+p = (m+p)+n$

Proof. Let $f(m,n,p) = (p=0 \rightarrow m+n, T \rightarrow f(m',n,p^-))$

Again f converges for all m,n,p. We have

$(m+n)+p = (p=0 \rightarrow m+n, T \rightarrow (m+n)'+p^-)$
$= (p=0 \rightarrow m+n, T \rightarrow (m'+n)+p^-)$ using Th 2.
$(m+p)+n = (p=0 \rightarrow m, T \rightarrow m'+p^-)+n$
$= (p=0 \rightarrow m+n, T \rightarrow (m'+p^-)+n)$

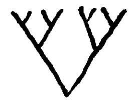
Each of these forms satisfies the equation for f(m,n,p).

Setting m=0 in Th 3 gives

$(0+n)+p = (0+p)+n$

so that if we had $0+m = m$ we would have commutativity of addition.

In fact, we cannot prove $0+m = m$ without making some assumptions that take into account that we are dealing with the integers. For suppose our space consisted of the vertices of the binary tree where m' is the vertex just above and to the left and m^- is the vertex just below and 0 is the bottom of the tree. m+n can be defined as above and of course satisfies Theorems 1, 2 and 3 but does not satisfy $0+m = m$.



We shall make the following assumptions:

1. $m' \neq 0$
2. $(m')^- = m$
3. $(m \neq 0) \supset ((m^-)' = m)$

which embody all of Peano's axioms except the induction axiom.

Th 4. $0+n = n$

Proof. Let $f(n) = (n=0 \rightarrow 0, T \rightarrow f(n^-)')$

$0+n = (n=0 \rightarrow 0, T \rightarrow 0'+n^-)$
$= (n=0 \rightarrow 0, T \rightarrow (0+n^-)')$
$n = (n=0 \rightarrow n, T \rightarrow n)$
$= (n=0 \rightarrow 0, T \rightarrow (n^-)')$ axiom 3

Th 5. $m+n = n+m$

Proof. By 3 and 4 as remarked above

Th 6. (m+n)+p = m+(n+p)
Proof. (m+n)+p = (m+p)+n   Th 3
= (p+m)+n   Th 5
= (p+n)+m   Th 3
= m+(n+p)   Th 5 twice

Th 7. m·0 = 0
Proof. m·0 = (0=0 → 0,T → m+n·0⁻)
= 0

Th 8. 0·n = 0
Proof. Let f(n) = (n=0 → 0,T → f(n⁻))
0·n = (n=0 → 0,T → 0+0·n⁻)
0 = (n=0 → 0,T → 0)

Th 9. mn' = m+mn
Proof. mn' = (n'= 0 → 0,T → m+m·(n')⁻)
= m+mn   axioms 1 and 2

Th 10. m(n+p) = mn+mp
Proof. Let f(m,n,p) = (p=0 → mn,T → f(m,n',p⁻))
m(n+p) = m(p=0 → n,T → n'+p⁻))
= (p=0 → mn,T → m(n'+p⁻))
mn+mp = mn+(p=0 → 0,T → m+mp⁻)
= (p=0 → mn+0,T → mn+(m+mp⁻))
= (p=0 → mn,T → (mn+m)+mp⁻)
= (p=0 → mn,T → mn'+mp⁻)

Now we shall give some examples of the application of recursion induction to proving theorems about functions of symbolic expressions. The rest of these proofs depend on an acquaintance with the LISP formalism
We start with the basic identities.
car [cons[x;y]] = x
cdr[cons[x;y]] = y
atom[x] [cons[car [x]:cdr[x] = x
atom[cons[x;y]]
null[x] = eq[x;NIL]

Let us define the **concatenation** x*y of two lists x and y by the formula
x*y = [null[x]→ y;T → cons [car[x];cdr[x]*y]]
Our first objective is to show that concatenation is associative.

Th 11. [x*y] *z = x*[y*z]
Proof •
nul [*y] = null [[null[x] → y;T → cons[car [x];
cdr[x *y]]]]
=[null[x] → null[y];T → null[cons [car
[x] ;cdr [x*y]]]
=[null[x] → null[y];T → F]
= null[x] null [y]
car[x*y] =[null[x] → car[y];T → car[cons[car
[x ; cdr[x]*y]]]
[null[x] → car [y] ;T → car [x]]
cdr[x*y] =[null[x] → cdr [y] ;T→ cdr [cons[car
[x] cdr [x]*y]]]
: [null[x] → cdr[y];T → cdr[x]*y]

Now
[x*y] *z =[null[x*y] → z;T → cons[car[x*y];
cdr [x*y]z
=[null[x] → [null[y] → z;T → cons
[car[x*y]; cdr [x*y]*z]] T→ cons
[car[x*y];cdr[x*y] *z]]
=[null[x] → [null[y] → z;T → cons [car
[y];cdr[y]*z T → cons [car[x];[cdr
[x]*y] *z]]]
=[null[x] → y*z;T → cons[car [x] [cdr
[x] *y]*z]]
Now let
f[x;y;z] =[null[x]→ y*z;T → cons [car[x];f[cdr
[x] ;y;z]]]
From the above steps we see that [x*y]*z satisfies the equation for f. On the other hand
x*[y*z] = [null[x]→ y*z;T → cons[car [x]; [cdr
[x]*[y*z]]]
satisfies the equation directly

Th 12. NIL*x = x
x*NIL = x
Proof. NIL*x = [null [NIL]→ x;T → cons [car[NIL];
cdr [NIL]*x]]
= x
x*NIL = [null [x]→ NIL;T → cons [car[x];
cdr [x] *NIL]]
Let f(x) = [null[x] → NIL;T → cons[car [x] ;f[cdr
[x]]]]
x*NIL satisfies this equation. We can also write for any list x
x = [null[x] → x;T → x]
= [null[x] → NIL;T → cons[car [x] ; cdr[x]]]
which also satisfies the equation.
Next we consider the function reverse x defined by
reverse [x] = [null [x] → NIL;T → reverse[cdr [x]]
*cons [car[x];NIL]]
It is not difficult to prove by recursion induction that
reverse[x*y] = reverse [y] * reverse [x]
and
reverse[reverse [x]] = x.

Many other elementary results in the elementary theory of numbers and in the elementary theory of symbolic expressions are provable in the same straightforward way as the above. In number theory one gets as far as the theorem that if a prime p divides ab, then it divides either a or b. However, to formulate the unique factorization theorem requires a notation for dealing with sets of integers. Wilson's theorem, a moderately deep result, can be expressed in this formalism but apparently cannot be proved by recursion induction.

One of the most immediate problems in extending this theory is to develop better techniques for proving that a recursively defined function converges. We hope to find some based on ambiguous functions. However, Godel's theorem disallows any hope that a complete set of such rules can be formed.

The relevance to a theory of computation of this excursion into number theory is that the mathematical problems involved in developing rules for proving the equivalence of algorithms. Recursion induction which was discovered by considering number theoretic problems turns out to be applicable without change to functions of symbolic expressions.

9. Relations to Other Formalisms

Our characterization of $C\{\mathcal{F}\}$ as the set of functions computable in terms of the base functions in $\mathcal{F}$ cannot be independently verified in general since there is no other concept with which it can be compared. However, it is not hard to show that all partial recursive functions in the sense of Church and Kleene are in $C\{\text{succ},\text{eq}\}$. In order to prove this we shall use the definition of partial recursive functions given by Davis[3]. If we modify definition 1.1 of page 41 of Davis[3] to omit reference to oracles we have the following: A function is partial recursive if it can be obtained by a finite number of applications of composition and minimalization beginning with the functions on the following list: 1)x', 2)$U_i^n(x_1,\ldots x_n) = x_i$, $1 \; i \; n$, 3)$x+y$, 4) $x \dot{-} y = (x-y > 0 \rightarrow x-y, T \rightarrow 0)$, 5)$xy$.

All the above functions are in $C\{\text{succ},\text{eq}\}$ Any $C\{\mathcal{F}\}$ is closed under composition so all that remains is to show that $C\{\text{succ},\text{eq}\}$ is closed under the mimalization operation. This operation is defined as follows: The operation of minimalization associates with each total function $f(y,x_1,\ldots x_n)$ the function $h(x_1,\ldots x_n)$ whose value for given $x_1,\ldots,x_n$ is the least y for which $f(y,x_1,\ldots,x_n) = 0$, and which is undefined if no such y exists. We have to show that if f is in $C\{\text{succ},\text{eq}\}$ so is h. But h may be defined by

$$h(x_1,\ldots x_n) = h_2(0,x_1,\ldots,x_n)$$

where

$$h_2(y,x_1,\ldots,x_n) = (f(y,x_1,\ldots x_n) = 0 \rightarrow y, T \rightarrow h_2(y',x_1,\ldots,x_n))$$

The converse statement that all functions in $C\{\text{succ},\text{eq}\}$ are partial recursive is presumably also true but not quite so easy to prove.

It is our opinion that the recursive function formalism based on conditional expressions presented in this paper is better than the formalisms which have heretofore been used in recursive function theory both for practical and theoretical purposes. First of all, particular functions in which one may be interested are more easily written down and the resulting expressions are briefer and more understandable. This has been observed in the cases we have looked at and there seems to be a fundamental reason why this is so. This is that both the original Church-Kleene formalism and the formalism using the minimalization operation use integer calculations to control the flow of the calculations. That this can be done is noteworthy, but controlling the flow in this way is less natural than using conditional expressions which control the flow directly.

A similar objection applies to basing the theory of computation on Turing machines. Turing machines are not conceptually different from the automatic computers in general use, but they are very poor in their control structure. Any programmer who has also had to write down Turing machines to compute functions will observe that one has to invent a few artifices and that constructing Turing machines is like programming. Of course, most of the theory of computability deals with questions which are not concerned with the particular ways computations are represented. It is sufficient that computable functions be represented somehow by symbolic expressions, e.g. numbers, and that functions computable in terms of given functions be somehow represented by expressions computable in terms of the expressions representing the original functions. However, a practical theory of computation must be applicable to particular algorithms. The same objection applies to basing a theory of computation on Markov's[9] normal algorithms as applies to basing it on properties of the integers; namely flow of control is described awkwardly.

The first attempt to give a formalism for describing computations that allows computations with entities from arbitrary spaces was made by A. P. Ershov[4]. However, computations with the symbolic expressions representing program steps are also necessarily involved.

We now discuss the relation between our formalism and computer programming languages. The formalism has been used as the basis for the LISP programming system for computing with symbolic expressions and has turned out to be quite practical for this kind of calculation. A particular advantage has been that it is easy to write recursive functions that transform program which makes generators easy to write.

The relation between recursive functions and the description of flow control by flow charts is described in Reference 7. An ALGOL program can be described by a recursive function provided we lump all the variables into a single state vector having all the variables as components. If the number of components is large and most of the operations performed involve only a few of them, it is necessary to have separate names for the components. This means that a programming language should include both recursive function definitions and ALGOL-like statements. However, a theory of computation certainly must have techniques for proving algorithms equivalent and so far it has seemed easier to develop proof techniques like recursion induction for recursive functions than for ALGOL-like programs.

References

1. Church, A., The Calculi of Lambda-Conversion, Annals of Mathematics Studies, no. 6, Princeton, 1941, Princeton University Press.
2. Church, A., Introduction to Mathematical Logic, Princeton, 1952, Princeton University Press.
3. Davis, M., Computability and Unsolvability, New York, 1958, McGraw-Hill.
4. Ershov, A.P., On Operator Algorithms (Russian), Doklady Akademii Nauk, vol. 122, no. 6, pp. 967-970.
5. Kleene, S.C., Recursive Predicates and Quantifiers, Transactions of the American Mathematical Society, vol. 53, 1953, p. 41
6. McCarthy, J., letter to the editor, Communications of the Association for Computing Machinery, vol. 2, August, 1959, p. 2.

7. McCarthy, J., Recursive Functions of Symbolic Expressions and Their Computation by Machine, Part I, Communications of the ACM, vol. 3, April, 1960, pp. 184-195.
8. McCarthy, J., The LISP Programmer's Manual, M.I.T. Computation Center, 1960.
9. Markov, A.A., Theory of Algorithms (Russian), Moscow, 1954, USSR Academy of Sciences, Steklov Mathematical Institute.
10. Naur, P., et al., Report on the Algorithmic Language ALGOL 60,Communications of the ACM, vol. 3, May 1960.
11. Turing, A.M., On Computable Numbers with an Application to the Entscheidungs Problem, Proceedings of the London Mathematical Society, ser. 2, vol. 43, 1937, p. 230; correction, ibid, vol. 43, 1937, p.544.

INFORMATION RETRIEVAL; STATE OF THE ART

Don R. Swanson

SUMMARY

Certain aspects of science communication are of especial importance to information retrieval. The exponential growth of science raises many questions related to increasing specialization. "Quality identification" is suggested as a critical issue. All approaches to information retrieval share a common set of basic problem areas and solutions. Semantics and redundancy are key conceptual issues and give rise to difficulties more likely to be overcome by meticulous thesaurus compilation than by any sudden insight or "breakthrough." The effectiveness of present retrieval capabilities is largely unknown, though certain recent studies are illuminating. Presentation and display to the user are suggested as important approaches to problems of information digestibility.

* * * * * *

Information retrieval embraces only one aspect of a broad class of science communication problems. A proper perspective for assessing the state of the information retrieval art can best be achieved through considering first the broader problem context. Certain elements of that context will be discussed before the subject of information retrieval itself.

Important Aspects of Science Communication

Who Pushed the Panic Button?

The obvious fact that the world's store of scientific information is increasing at an exponential rate has apparently enjoyed independent discovery by scores of science information writers during the past few years. On a suitably scaled graph, any rising exponential curve tends to produce an hypnotic effect of impending crisis. Now an atmosphere of alarm, since it is conducive to action, is no doubt beneficial, but only provided such action is properly directed. Increasing volume of information is not in itself the real problem. It can be inferred in fact that in the context of increasing world scientific population, the information increase rate is not at all out of balance. The number of published papers, the number of journals, the number of scientists, and expenditures for research, all have been increasing at an annual rate of 5-7% for the past 250 years.[1] Since the earth's population increases at only 1.6% annually, indiscriminate extrapolation of the exponential growth of science clearly is absurd.[11] Limiting factors will of course set in. Prof. Bar-Hillel[2] has challenged the view that anything at all is worsening (so far as basic long term trends are concerned), and rightly insists that the prophets of calamity must produce more convincing evidence than that which has yet come to light. Since on a scientist per capita basis the quantity of scientific information remains about constant, the case for pushing a panic button is at best obscure. The real issue must be identified as a changing partition of an increasing supply of human knowledge among the increasing number of scientists who generate and assimilate that knowledge. More simply, a scientist cannot now realistically expect to keep on top of as large a portion of any particular field or discipline (such as "physics" or "logic") as he could some years ago. The problem must be considered in terms of "backlog" as well as "current production."

Many profound questions, presently unanswerable, should be diligently explored, not in an atmosphere of crisis but in recognition of their long term fundamental importance. Should scientific journals be organized to meet increasingly more specialized needs? Does the manner of journal organization itself influence the course of scientific research? Can excessive specialization inhibit scientific creativity? Should steps be taken to "purge" the world's scientific literature, or at least to separate the important from the unimportant? Are more surveys, review papers, and monographs needed? How should education systems and information retrieval systems be kept flexibly responsive to the increasing segmentation of scientific knowledge?

These and other issues must be identified so that guidelines to future research may be developed but further pursuit of the matter is inappropriate here. So far as information retrieval itself is concerned, it will be assumed from this point on that things are bad enough and ought to be improved, regardless of the rate at which they may or may not be getting worse. The conceptual, rather than equipment, aspects of the subject will be emphasized.

Mooers' Law

Mooers[3] has postulated that "an information system will tend not to be used whenever it is more painful and troublesome for a customer to have information than for him not to have it." By way of explanation, he observes that the penalties for not being diligent in library research are minor if they exist at all. Clearly a greater amount of recognizable and rewarding (even though duplicative) work can be accomplished if time is not spent in the library. Effective and efficient information retrieval systems have been installed and then removed for lack of use; the customer's ability to retrieve information outstripped his motivation. It is suggested in effect that the diligent finding and use of information should be rewarded and failure to do so punished; in these circumstances we can expect better retrieval systems.

Though Mooers may have overstated the case, his observations are provocative. If this state of affairs indeed exists, and certainly it must to some extent, then the question of remedial action is difficult and important. In addition to the concepts of "reward" and "punishment", two additional directions for improving "motivation" seem to be promising.

First, more attention ought to be paid to active dissemination of scientific information rather than passive interment in libraries wherein the resurrection of information requires initiative and ingenuity on the part of the customer. Perhaps the scientific community is in need of something akin to "screening panels" whose job is to direct newly acquired information to the appropriate potentially interested users.

Secondly it is suggested that beyond, after, and independently of "information retrieval" per se, the proper presentation and display of information to the customer may play an important role in the palatability and digestibility of the retrieved information. The extension of present information retrieval boundaries in this direction will be discussed later in the framework of research trends and goals.

Birth Control of Technical Reports

Scientists are painfully aware that much, if not most, scientific literature is either so repetitive, verbose, or of such low quality, that it ought not to have been written in the first place. In the opening address at the 1958 International Conference on Scientific Information, Sir Lindor Brown made a sparkling appeal for the exercise of restraint on the part of both authors and publishers.[4] In professional journals, quality of information is at least partly controlled through editorial sanction, but the now comparable volume of unpublished technical reports are totally immune to such controlling influence. Dwight Gray[5] pinpoints the lack of bibliographic control as an especially unfortunate aspect of technical reports.

It is doubtful that any system of "information birth control" can be made practical, particularly one that appeals to voluntary self restraint; involuntary control however may be a cure worse than the disease. "Quality Control" applied more liberally may be a part of the answer, but any connotation of "control" raises the question of information suppression without competent and diligent exercise of the judgmental function. "Quality Identification" is suggested here as a more appealing approach. Clearly in a situation in which the volume of information exceeds the individual scientist's digestive capacity, some method of assisting him to discriminate between the important and the unimportant literature can be counted among the most crucial of requirements.

The referee system for professional journals imposes a modest degree of quality control on scientific information. It is at least thinkable that recognized leaders in the various technical fields could be imposed upon to implement a "quality identification" system on a sizable scale. However, a more practical technique immediately at hand may lie in the idea of "citation indexing." It is not unreasonable to measure the importance of a published article or technical report by the frequency of subsequent citations to that article or report (possibly taking into account who did the citing and why.) This suggestion was ably described by Garfield in an article published six years ago.[6] Experimental studies of citation patterns in physics literature has been carried out by Kessler.[7]

The idea of automatic abstracting has enjoyed considerable publicity and popularity during recent years as an approach to information compression. Techniques for so doing have somehow managed to come into being without the benefit of any adequate description of requirements. The situation can be rather easily understood by first considering one plausible elementary rule for "automatic abstracting" (or rather "automatic extracting"): "Select title and first sentence of each paragraph." This rule is obviously mechanizable. Furthermore, at the time of this writing, no other rules have been demonstrated on the basis of which a convincingly better abstract can be produced. The problem lies in the fact that adequate measures for the "quality" of an abstract (human or machine) have never been developed. Ideally, of course, one would like to cut the length of an article drastically without appreciable loss of information, but unfortunately it cannot yet be demonstrated that the percentage of information loss is significantly smaller than the percentage of text reduction. The question of a need for text reduction even at the expense of information loss is nonetheless open.

An interesting and constructive approach to "volume reduction" is contained in a recent Case Institute report,[8] a part of which concerns the effect of condensation on reader comprehension of journal articles. If indeed one can somehow single out "measurable" factors in the area of reader comprehension, then certainly experimental observations of this kind are crucial to questions of redundancy elimination and abstracting.

"Communication Habits of Scientists"

The scientific community itself has been the object of attention of a number of "opinion surveys" and scientific communication "behavior studies." These studies represent a commendable attempt to describe characteristics and attributes of our present system of scientific communication. A comprehensive review of several dozen such studies has been reported by Menzel.[9] The stated purpose of this review is to display the variety of research that has been done on the flow of information among scientists; it does not attempt, however, to evaluate or recapitulate that which is worthwhile and deserving of future emphasis.

The total amount of information of this kind presently available tends to cause mild indigestion when one attempts to assess the relevance of the whole matter to the implementation of practical improvements in scientific communication. In their present state, however, these "behavior" studies are intended only to be descriptive rather than diagnostic. It is premature to attempt to translate a description of present behavior into future requirements; to discern any relationship between the two requires considerable further study.

Certain fragmentary results of a rather general nature are worth noting, however, since they have an interesting bearing upon some of the questions raised here. Let us consider a few excerpts from the Menzel review.

"The amount of time chemists devote to reading on the job was found directly related to the ease of their access to scientific literature. Reading time was directly related to the availability of journals at chemists' desks, to the location of library facilities in the chemists' building, and to the existence of company library facilities."

"'one of the greatest stimulants to the use of information is familiarity with its source.'"

"Sometimes pieces of work which have been ignored by the scientific community prove to be highly significant when someone finally stumbles upon them in the back volumes.....It is suggested tentatively that it is often necessary to publicize information repeatedly, lest it fail to enter the stream of communications which will lead to its ultimate user. From the point of view of the consumer of the information, it seems sometimes necessary to be exposed to the information repeatedly before it will make an impact."

"On the basis of the information on chemists' reading time and on the number of articles abstracted in Chemical Abstracts in a given year, it was concluded that only one half of one percent of the articles published in chemistry are read by any one chemist."

"a large portion of articles read and considered useful have been met with by chance;"

"'there is thus a good deal of circumstantial evidence for the hypothesis that the literature is used very much more for news than for reference.'"

The foregoing conclusions are provocative and clearly have some bearing on the issue of "literature search" versus "specific-information retrieval," as well as on the matter of Mooers' Law. It should be realized that the results as they stand shed no light on cause-effect relationships. It is not clear whether the use of current literature for news rather than reference reflects a requirement or whether it simply reflects the fact that current literature information systems are inadequate for reference purposes, and hence can be used only for news.

This identification of problem areas at least suggests action in two directions. First since the stimulant effect of scientific publication is known to be important, steps to enhance such stimulant capability can be taken. If this idea were coupled with "quality identification" we might reasonably equate "quality" with "stimulation ability." Once stimulating material has been identified, provision could then be made for wider circulation within the scientific community. The conclusion that current literature is not really used very much for reference, if correct, is dramatic in its implication of a requirement for a profound look at and perhaps overhaul of our literature reference systems.

The spectrum of behavior studies reviewed by Menzel does not apparently include any reports of active controlled experiments which exert a perturbing influence on the scientific community. Experiments with a fully controlled model system have been suggested by Kessler,[7,10] and seem worthy of serious consideration.

The Conceptual Nature of Information Retrieval

All approaches to information retrieval are reducible to a common set of elements, both from the point of view of problems and solutions. The objective of any information retrieval system is to permit the originator of information to communicate with an unknown user at some unknown future time. We may presume that the user or customer is faced with a large volume of information, i.e., a library which by many orders of magnitude is too large to permit an exhaustive direct examination search to meet whatever requirements for information he might have. Thus communication from originator to "user" must take place via some abbreviated "representation" of the contents of the library. This representation may take many forms, for example: classification systems, alphabetic subject indexes, coordinate systems, facet classification systems, full text, and others yet to be invented. They all incorporate certain common problems, and these problems have a common origin.

When any document is indexed, cataloged, classified, or otherwise tagged, this process constitutes an attempt to represent "information content" of the article or document in a highly abbreviated form. This "representation" is intended to serve the purpose of information retrieval but contains only a minute fraction of the information contained in the document. Now it is not usually supposed that the "representation" should reflect the total "information content", yet at the same time there is no solid theoretical reason that the purpose of information retrieval can be fully served in any other way. Fortunately, the whole matter need not rest on theoretical proof since the objective at hand is of a purely practical nature.

As a practical matter, the encoding of "information content" or "meaning" of a document in any highly abbreviated representation is an intuitive process not amenable to formal description. Experiments have indicated that the reproducibility of the intuitive judgment of indexers or catalogers is relatively low. A subject heading, index term, or descriptor, has a "meaning" which is a function of the viewpoint of the observer or equivalently a function of the context in which that heading, term, or descriptor is utilized.

The effectiveness of an information retrieval system depends on the success with which the indexer and user can pursue the following strategy. In creating an "abbreviated representation" of an article being indexed, an attempt is made by the indexer to foresee essentially all ways in which some unknown user might wish to recover the information. A rather high degree of redundancy is a result of a deliberate attempt on the part of the indexer to predict the customer's viewpoint. Similarly the search is conducted on a redundant basis since in effect the user hunts in all likely (and in some not-so-likely) places. Both indexer and user must therefore play a "guessing game" to ascertain the viewpoint of the other. This game results in redundancy, but not so much so that large quantitites of retrieved irrelevant information thwart the purpose of the search. The foregoing description applies to a successful information retrieval system. The fact is, it is possible to find many systems which seem to have no elements of such guessing or redundancy and which, at the same time, are not very successful.

Consider for the moment a retrieval system in which the library "representation" is based solely on key words or descriptors. For this situation, the "guessing game", played by both the indexer and searcher, can be aided by means of a thesaurus. For our purpose here, we define a thesaurus to be a collection of groups of words wherein the various members of a group tend to mean about the same thing for purposes of information retrieval. These word groups thus serve as a reminder list to assist the indexer and searcher in conjecturing how the same idea might be expressed in many ways. (Experimental use of a thesaurus is described for full text search in references 11, 12.) In one form or another, something akin to a thesaurus must be implicitly present in any good retrieval system. The "see also" portion of an alphabetic subject index or of a classification system can be looked upon as serving the purpose of a thesaurus.

The total amount of desirable redundancy in any retrieval system, and the detailed manner in which redundancy should be distributed between indexing and searching, depends upon economics and on the question of irrelevant retrieval. For a system in which the encoding process is highly efficient, redundancy may be cheaper to come by as a part of the input rather than as part of the output. In other systems, however, such as those in which the search workload is light, it may be that redundancy can be purchased more cheaply in the searching process.

An important problem area, more or less distinct from questions of meaning, has not been covered in the foregoing discussion. This area concerns syntactic relationships among the various index terms that might be assigned to a given document. The effect of failure to utilize such syntactic relationships is to increase the retrieval of irrelevant material but with no loss of that which is relevant. To illustrate, if the phrase "the effect of radiation on mutation" is an appropriate description of the content of an article, then the independent assignment of descriptors corresponding to "radiation" and "mutation" would not by itself preserve the syntactic relationships between these two terms implied by the original phrase. Irrelevant retrieval of documents in which the descriptors "mutation" and "radiation" were simply included among a number of other descriptors in such a way that those two were not related to one another, could clearly result. The degree to which this whole question is of importance in terms of the amount of irrelevant retrieval that might be caused by ignoring syntax has not been experimentally established. In principle though the problem is of considerable importance and a number of approaches to its solution should be mentioned.

First of all, a considerable amount of work has been in progress for some years at the University of Pennsylvania under Z. Harris.[13] The objectives of this effort are broader and more fundamental than the question of preserving syntactic relationships among index terms, but the subject is nonetheless of considerable relevance. Eugene Wall[14,15] has presented a lucid categorization of information retrieval problems in terms of viewpoint, generics, semantics, and syntax. He describes the use of "role indicators", as a solution to problems of syntax, and claims that effective and successful technical information systems which employ only twelve such indicators have been in operation.

So far as full text searching is concerned, an especially simple substitute for syntax, namely, the "proximity" of terms has been suggested by the author[12]. This approach depends on the fact that two terms which appear in the text of an article in reasonable proximity to one another (i.e., within a few sentences) have a high probability of being syntactically related.

"Effectiveness" and "economy" can be identified as the two fundamental objectives of information retrieval research. "Effectiveness" has to do with how well a system works, in terms of both the percentage of relevant material retrieved and the accompanying amount of irrelevant material. "Economy" of course refers to the cost of operating the system, including indexing, storage, file maintenance, and searching. In particular, the "cost" to the user of reading irrelevant material must be included. (Otherwise the distinction is subject to total confusion by the existence of a hypothetical library system of 100% effectiveness and essentially zero cost. This library is an unorganized, unindexed, unmanned warehouse which the customer reads through from beginning to end for each information request.) Any research project on information retrieval should be assessed in the light of its objectives, and the division of objectives into the two categories of effectiveness and economy usually provides a perspective not otherwise apparent.

The issue of mechanization is, of course, tied to economy. It is obvious, though sometimes overlooked, that computing machines cannot in principle improve the "effectiveness" of anything at all over what can be done with people, unless response times or environmental factors are of importance. The basic question is one of economy alone. Speed itself may be of some relevance but, in principle at least, one can duplicate the speed of any automatic process with enough people working in parallel. Admittedly this argument is oversimplified (though the remark on response times and environment is a strong hedge), but justified as a counteraction to the unfortunately prevalent belief that mechanization of a process that doesn't work to begin with will improve matters.

Apart from its intimate relationship to economy, one other especially important factor in mechanized information retrieval should be recognized. The machine handling of the physical contents of a library involves problems totally different from the machine handling of a representation which permits that library to be searched. Present general purpose computing equipment, provided it is used with some ingenuity, is not badly suited to machine search of representations, i.e., index and catalog type information. So far as handling the total information content of a library is concerned, the required storage jumps several orders of magnitude, and existing computers are largely inappropriate. The output of computers which handle and search only an index-representation is necessarily just a bibliographic listing of responsive documents (accompanied possibly by titles or brief abstracts). Retrieval of the document itself (from shelves, filing cabinets, or microfilm storage) must then follow as a second stage process. Special purpose retrieval machinery can be relatively efficient for this second stage, though generally speaking well designed manual filing systems offer serious economic competition.

Some equipment designs have been based on a merged record (usually recorded on film) of the machine-readable coded representation of a document with a non-machine-readable full text document image. This merger imposes awkward machine design problems from the beginning and must depend for its final justification on overall economy of the resulting system. Up to the present time large scale systems based on the merger philosophy have not been conspicuously noted for economy and simplicity (to risk a serious understatement). (That fact was of course known from the design stage on--what's worse though, at least some of these systems after installation have been victims of a rather severe onset of Mooers' Law). The question of mechanization in libraries is full of economic pitfalls, but not to the extent of preventing soundly designed systemsto be implemented with present hardware. Since this paper is intended to focus on conceptual problems, the matter of equipment will not be further pursued.

A considerable amount of reported work in the information retrieval field is indirectly addressed to the question of economy rather than effectiveness though not in an obvious sense and not necessarily in connection with mechanization. A number of ingenious logical-mathematical models of information retrieval systems have been devised. Many of these papers have interesting implications on questions of file organization and search strategies, (e.g. Estrin[18], Moore[19]) and possibly on the design of future machines; their promise for leading to new insights with respect to the more basic semantic problems that lie at the core of "retrieval effectiveness" is less clear.

It is at least plausible, however, that some of the mathematical models might eventually lead to fruitful results in a more fundamental sense. Mooers[20] has developed a model in which certain mathematical properties of different types of retrieval systems can be compared. It would seem useful to extend a model of this kind to include formal representation of redundancy, and then to investigate the relationship between retrieval effectiveness and redundancy. In the terminology of Mooers' model, the transformation relationships between the space of all retrieval prescriptions and the space of all document subsets should be formulatable in such a way that the effect of redundancy is clearly brought out. It would be overly bold to predict at this point whether such an approach would yield any new fundamental insights, better file organization and storage strategies, both, or neither.

A key element in understanding the "nature of information retrieval" must certainly be the types of questions posed by users of information systems. It is in fact far more reasonable to design library representations on the basis of the way in which the users tend to organize the subject matter rather than the way in which indexers imagine that it ought to be organized, yet it seems that this procedure is seldom followed. This approach is embodied in a study carried out by Herner[21] on the information system of the U.S. Atomic Energy Commission. (Several interesting results with immediate practical implications, so far as machine design is concerned, were obtained. Ninety percent of the questions involved three or fewer distinct concepts and of all of the multiple concept questions, ninety-eight percent involved logical products rather than logical sums or differences.) In connection with the Herner study, it would be interesting to know the extent to which the user's questions were predicated on some prior notion of how the AEC library was organized; that is to say, did their questions really reflect what they wanted or were they conditioned by what they thought the library could provide?

How Effective are Present Information Retrieval Systems?

It is a curious fact that the above question is essentially unanswerable in terms of any objective "measurables." For any given information retrieval system, those concerned with it are generally not at a loss for an opinion as to how well it works, but such opinions are seldom accompanied by evidence. Some recent experimental research has been seriously addressed to this point, and insight is being acquired. Large scale experiments which attempt to measure retrieval effectiveness and at the same time compare various factors crucial to such effectiveness, are presently under way (under the direction of Cleverdon) at the College of Aeronautics, Cranfield, England. Preliminary results have been reported.[16] Within the framework of a small scale experimental system, considerable attention is given to the question of defining measures for retrieval effectiveness in the author's investigation of text searching.[11] The results of several retrieval methods are compared with "direct examination" of the whole document collection.

The emerging results reported in these two recent studies seem to be taking on a rather interesting pattern. The ASLIB Cranfield project had for its objective the measurement of three variables: the indexing system (four specific systems were compared), the time spent indexing, and the experience or qualifications of the indexer. Preliminary results presented by Cleverdon in a recent seminar[17] indicate that it doesn't much matter which indexing system is used, how much time is taken in the indexing process, or whether the indexer did or didn't have a lot of experience. Nothing seemed to depend critically on the searcher either. Similar "invariances" were encountered in the reported text searching experiments.[11]

On the whole retrieval effectiveness in these experimental systems was relatively low (Cleverdon reports recoveries in the neighborhood of eighty percent but it should be noted that these figures refer to "source" documents--i.e., those which directly inspire the retrieval question. Retrieval percentages for non-source documents have not yet been reported for that project. The same distinction between source and non-source documents was used by the author,[11] and it was found that recovery of source documents was about twice as effective as that for non-source documents.) In general these experiments seem to indicate mediocrity of retrieval effectiveness and insensitivity to parameters that might reasonably be supposed important. Though further analysis is necessary before firm conclusions can be drawn, certain hypotheses to explain the observed mediocrity and insensitivity can be suggested. If we suppose that the thesaurus used by the author and a "see also" structure of the indexing systems tested by Cleverdon were about equally primitive (insofar as covering any substantial range of language redundancy is concerned) then one would have expected the results to come out about as they did. It has already been found that a substantial portion of the text searching ineffectiveness could reasonably be attributed to a deficient thesaurus.[12]

An interesting sidelight brought out by Cleverdon in the NSF Seminar is that there presently exists no book or well organized doctrine on how to compile subject heading lists nor does there exist a systematic body of opinion on the application of "see also" references. Cleverdon observes that the latter particularly tend to be applied in a haphazard manner. With inadequate thesaurus groups, cross references, and haphazard "see also" application, it should come as no surprise that the indexer and user find difficulty in communicating via an information retrieval system.

A few remarks can be made on the state of automatic indexing. It has been pointed out that the language redundancy problems associated with retrieval effectiveness must be approached through thesaurus-like compilation techniques. On the basis of all present evidence, there is no particular reason to believe that the process is any more difficult for automatic indexing than it is for human indexing. The more subtle aspects of semantics and redundancy in a system based on automatic indexing must in that case be thrown into the search process, but with no obvious disadvantages. The real question is not so much whether automatic indexing can be made to work (though to be sure the matter must still be left open) but whether it can be made economical. Generally speaking, with present computing equipment it cannot, except for certain special applications where input costs can be amortized in other ways. Engineering achievements in the area of direct printed page input and higher speed memory readout, may hold the final answer to this question.

Trends and Goals

The nature of the basic problems of information retrieval is such that no sudden conceptual breakthrough seems likely. Current inventories of mathematical theories and techniques are applicable to information systems only to a limited extent. The tasks that clearly lie ahead must include large scale language studies and laborious experimental investigations. The past several years have seen increasing awareness of the significance of some kind of thesaurus approach to problems of multiple meaning, viewpoint, generics, and redundancy. This approach is not limited to natural language search techniques or to key-word descriptors, but finds its counterpart in the "see also" cross references of all types of subject headings and classification systems. These compilation efforts which in effect attempt to build "structures of relatedness" in indexing and retrieval systems should be closely tied to studies of the types and forms of questions which users ask of the system. Studies of users' questions should be patterned not only after the kind that are presently asked (e.g., Herner[21]) but should be addressed also to the kinds of questions which users ought to ask if they had different and better information retrieval systems.

Retrieval techniques work best when they deal with a limited subject area and are tailored to the requirements of a limited group of users. Working systems of this kind should form the nucleus for experimental investigations so that deeper insights can be obtained into the question of how and why they behave as they do.

Syntactic studies should continue though it may be anticipated that their practical relevance to problems of information retrieval may not materialize to the degree that many workers presently hope. It seems to the author that problems of semantics tend to dominate practical requirements. It is a more or less obvious phenomenon of language that essentially the same concept can be expressed in many dozens of ways which are not syntactic transformations of one another and this fact alone suggests that syntacticians may be eventually disappointed in the extent to which their work finds practical application. Storage and retrieval of condensed or "kernelized" sentences[13] suffers from the same illness as do other methods of automatic abstracting; at present there are no acceptable measures for the amount of information loss in the kernelization process. This entire area may well emerge as being of considerable future importance however if for no other reason than the keen interest of a considerable number of competent workers; it holds much potential for taking off in new directions.

The presently primitive state of the automatic indexing art was mentioned briefly in the last section, and it is clear that studies of this kind shall and should continue. It is in this area that advances in equipment capabilities are urgently needed. A portion of these studies should be addressed to the interplay between natural language queries and mechanized information retrieval systems. The formulation of a request in natural language for computer processing is not subject to the current limitations of high volume storage that natural language text searching involves.

The extent to which one should be sanguine about continued studies of the information gathering habits of scientists, as though they were a colony of bees or ants, is not really clear. Certainly a rash of these studies has broken out during the past four years or so and many of the results have been interesting. It is possible that these have largely run their course and that now emphasis ought to shift to experimental studies that are subject to greater control. Certainly more investigation should be made of the obviously observable parameters within the science communication sphere such as those provided by the technique of citation indexing. Other techniques exist for "hooking" documents together by purely pragmatic and intuitive estimates of relevance within the framework of some particular purpose. A set of citations as useful "relevance hooks" would be based on an assumption that co-citation implies similarity of meaning from the point of view of the author doing the citing. In an analagous sense, Fano[22] suggests that the recognition of similarity of two documents in the process of request formulation (i.e., the requestor asks for a document "like" one he already has) provides relevance hooks similar to those provided by co-citation. Networks of relevance chains once assigned are susceptible to highly efficient machine manipulation and have the added virtue that the profound questions of semantics, viewpoint, ambiguity, generics, and syntax, are involved only to the extent of directly observable external effects resulting from a large collection of instances in which professional judgment is applied to the question of "relatedness."

It was mentioned earlier in connection with Mooers' Law that extending the boundaries of information retrieval into the area of presentation and display may provide an approach to solving problems of information indigestion on the part of the user. Strictly speaking, an information retrieval system has served its purpose once responsive documents have been delivered to the customer, but if indeed the customer's attitude is characterized by indifference, to the extent that the retrieval system is not effectively used, then we might surmise that a highly important and critical problem area has been ignored. This area is one which begins where information retrieval ends. Let us imagine that a stack of several dozen documents of several thousand words each has been retrieved. Now the customer's original motive in requesting those documents must be examined. If he seeks specific pieces of related information, a considerable effort on his part may be required to extract what he wants. If that is indeed the case then further machine processing of such retrieved data may serve a useful purpose. It is of particular significance to note that computer handling of the total retrieved text (in contrast to the total text of the library) is reasonable in terms of storage requirements. His original requirement may of course have been of a different kind, such as acquiring general familiarity with a subject area, but we shall consider further only that situation which calls for specific small fragments of information to be brought together and "correlated" in some abstract intellectual way. This type of requirement typically occurs in a business or military intelligence application. The question now is whether or not the proper presentation, rearrangement, and display of numerous fragments of retrieved data may play a significant role in stimulating the user to perceive relationships that are not otherwise apparent in a more laborious process of directly examining the total contents of the retrieved material. Experiments carried out by the author and co-workers at Ramo-Wooldridge during the last year or so have demonstrated that such stimulation is indeed possible. (A report on this work is in preparation.) When combined with one of the conclusions quoted in the Menzel review,[9] namely that the periodical literature tends to be used more for "idea stimulation" than for reference, these concepts of presentation and display take on added significance. It is suggested that future information retrieval work may exhibit an interesting trend in the direction of autmatic user stimulation.

The articles cited in the following list of references themselves contain references to 160 other articles (probably some are duplicative). If succeeding generations of citations are counted, one wonders whether a closed system will be encountered (it seems likely) and whether subsets of those cited many times are in some way more "central" or "important" to information retrieval.

REFERENCES

1. Scientific and Technical Information as One of the Problems of Cybernetics by G.E.Vleduts, V.V.Nalimov, N.I.Styazhkin; SOVIET PHYSICS USPEKHI, Vol. 2 No. 5, p 637, Sept.-Oct. 1959

2. The Real Problems Behind Information Retrieval and Mechanical Translation, Y.Bar-Hillel, Seminar Feb. 13, 1961, Sponsored by the Office of Naval Research

3. Information Retrieval Selection Study, Part II: Seven Retrieval System Models, by Calvin N. Mooers, Zator Company, Report No. RADC-TR-59-173 Contract AF30(602)-1900

4. Opening Address, Sir Lindor Brown, ICSI* p.3

5. Technical Reports I Have Known, and Probably Written, by Dwight E. Gray, Physics Today, V.13 No. 11, p.24 Nov.1960

6. Citation Indexes for Science by Eugene Garfield, Science, V.122 No. 3159 p.108 July 15,'55

7. Technical Information Flow Patterns, M. M. Kessler, WJCC May 1961

8. An Operations Research Study of the Dissemination and Use of Recorded Information by Operations Research Group, Case Institute of Technology, December 1960 (Sponsored by National Science Foundation)

9. Review of Studies in the Flow of Information Among Scientists, Bureau of Applied Social Research, Columbia University, January 1960 (Sponsored by National Science Foundation)

10. Background to Scientific Communication by M. M. Kessler, IRE Translations, Vol. EWS-3 No. 1 April 1960

11. Searching Natural Language Text by Computer, by Don R. Swanson, Science, Vol. 132 No. 3434 p.1099 October 21, 1960 (Sponsored by Council on Library Resources)

12. Research Procedures for Automatic Indexing by Don R. Swanson, presented at American University Third Institute on Information Storage and Retrieval, Feb. 16, 1961, (Sponsored by Council on Library Resources)

13. Linguistic Transformations for Information Retrieval by Z. S. Harris, ICSI* p. 937

14. Documentation,Indexing,and Retrieval of Scientific Information, prepared by Committee on Government Operations, U.S.Senate, Doc. No.113 (report by Eugene Wall p. 175-202)

15. Summary of Area 4 Discussion, ICSI* p. 804,805

16. ASLIB CRANFIELD RESEARCH PROJECT, Report on the First Stage of an Investigation Into the Comparative Efficiency of Indexing Systems by Cyril W. Cleverdon, Sept. 1960 (Sponsored by the National Science Foundation)

17. Seminar February 17, 1961, by Cyril W. Cleverdon, held at National Science Foundation, Washington, D.C.

18. Maze Structure and Information Retrieval, Gerald Estrin, ICSI* p. 1383

19. A Screening Method of Large Information for Retrieval Systems, Robert T. Moore, WJCC May 1961

20. A Mathematical Theory of Language Symbols in Retrieval, by Calvin N. Mooers, ICSI* p.1327

21. Determining Requirements for Atomic Energy Information from Reference Questions, by Saul Herner and Mary Herner, ICSI* p.181

22. Summary of Area 6 Discussion, ICSI* p.1407

*ICSI: Proceedings of the International Conference on Scientific Information National Academy of Science, National Research Council, Washington, D.C. 1959

TECHNICAL INFORMATION FLOW PATTERNS

M. M. Kessler
Lincoln Laboratory,* Massachusetts Institute of Technology

Summary

A study of the bibliographies of a large number of articles in physics and electrical engineering indicates that definite patterns exist for the flow of technical information. Quantitative data are presented on the flow of information between countries, between cultural and functional groups, and between past and present. An analysis of the numerical data indicates that these flow patterns are deeply rooted in the dynamics and evolution of scientific thought and engineering development. The analysis also discloses that extreme asymmetry exists between journals in their capacity as carriers of scientific information.

Introduction

Every project in the field of information retrieval must face up to the following question: "Will it promote the flow of meaningful information from originator to consumer?" No idea, scheme or component, no matter how intellectually clever or technically elegant, has any worth except insofar as it contributes to an information flow system. This flow system has its carriers, channels, sources, and sinks. It defines the coupling between individual scientists across field boundaries, political groupings, time, and habits of tradition. It is clear that any new scheme or component of information retrieval, in order to be effective, must mesh into the flow pattern and be properly matched to it. And yet we know remarkably little about the information circuits as they now exist.[1] This paper attempts to map the flow of information by analyzing the statistics of references that authors include in their published papers. It is assumed that the published journal article is the message unit and that its citation by an author is recorded evidence that the message has found a meaningful target. The strength of this method is that within its area of limitations it is quantitative and unambiguous. It does not depend on subjective opinions and questioners, and very significantly, a wealth of recorded data exists in journals of all countries, all sciences, and as far into the past as we care to go. The weakness of the method is that it certainly does not measure all scientific communication. Other modes and circuits exist that are not reflected in bibliographic citations and conversely some citations may be irrelevant to the information transfer process. The method is nevertheless particularly applicable to the problems of retrieval systems because the latter is largely concerned with communication through written papers.

This study is concerned with the literature of physics and closely related fields of application. Data are presented on the flow of information across political boundaries, from a basic science to an applied technology, from past to present, and concludes with some remarks on the efficiency of various journals as carriers of information. The results are then discussed in terms of their application to the design of a system for scientific retrieval and communication.

Numerical Data

Communication across Political Boundaries

A number of journals were analyzed for purpose of obtaining a rough measure of the flow of scientific information across national boundaries. The bibliographies in the indicated journa were sorted on the basis of country of origin. Table I shows the results for the January 1957 issue of the Physical Review.

Table I

Geographic Distribution of References in the Physical Review (January 1957)

Reference to	No. of References	% of Total
Physical Review	994	48.0
Other American	558	27.0
British	198	9.5
European	258	12.4
Russian	28	1.4
All others	33	1.6

We see that roughly half of the references in the Physical Review are to papers published in the same journal. Three quarters of all the references are to American journals. The remaining 25% of the references are distributed among a variety of European journals, mainly British. Note in particular the vanishing call on Russian reference material (1.4%).

* Operated with support from the U. S. Army, Navy and Air Force

We now consider the same process as it operates on Russian physicists. Table II gives the results of a count on the June and October 1957 issues of the Journal of Theoretical and Experimental Physics.

Table II

Geographic Distribution of References in the Russian Journal of Theoretical and Experimental Physics (June and October 1957)

Reference to	No. of References	% of Total
JETP	102	15.4
Other Russian	201	30.5
Physical Review	148	22.4
Other American	57	8.7
British	63	9.5
European	66	10.1
All Others	22	3.3

Note that on its home grounds the Russian JETP actually runs second to the Physical Review. The Russian physicists depend on British and other European literature roughly to the same extent as do the Americans, but they do not have the strong partiality to their own chief journal nor to any combination of journals in their political group.

Similar counts were made on Nuovo Cimento and Physica, Italian and Dutch journals of physics. The results are shown in Tables III and IV.

Table III

Geographic Distribution of References in the Italian Nuovo Cimento (Jan. - June 1958)

Reference to	No. of References	% of Total
Nuovo Cimento	344	15.7
Other Italian	38	1.7
Russian	84	3.9
Physical Review	771	35.6
Other American	331	15.4
British	223	10.5
European	245	11.2
Others	135	6.2

Table IV

Geographic Distribution of References in the Physica (1957)

Reference to	No. of References	% of Total
Physica	225	21.6
Other Netherlands	71	5.9
Russian	30	3.0
Physical Review	249	23.9
Other American	85	8.2
British	159	15.2
European	121	11.6
Others	108	10.4

In view of the political and cultural polarization of modern society into East and West groupings, it is interesting to present the data of Tables I to IV in such a way as to illustrate the flow of physics from East to West and vice versa. Table V groups all references to American and European journals and compares them with those to Russian and other "iron curtain" journals.

Table V

The Flow of Information along the East-West Political Axis

Articles in / References to	Phys. Rev.	Nuovo Cimento	Physica	JETP
Western Jnls.	96.9	90.1	86.4	50.7
Eastern Jnls.	1.4	3.9	3.0	45.9
All Others	1.6	6.2	10.4	3.3

To extend this picture somewhat beyond pure physics, a count was made on the Journal of Applied Physics (JAP) and the Proceedings of the Institute of Radio Engineers (IRE). In the latter case, we picked January, June, and September 1957 as representative issues. A special issue of the IRE devoted to a symposium on transistor technology was treated separately. The results are shown in Table VI. The Phys. Rev. data are reproduced for comparison.

Table VI

Geographic Distribution of References in Phys. Rev., JAP, and IRE

Articles in / References to	Phys. Rev.	Jn. App. Phys.	Proc. IRE	Proc. IRE Transistor Issue
U.S.A.	75.0	70.5	77.0	78.0
British	9.5	12.0	11.6	5.3
European	12.4	11.7	8.7	8.6
Russian	1.4	2.9	1.2	2.4
Others	1.6	2.9	1.4	1.9

The data suggest the following conclusions.

1. The Physical Review is truly a definitive journal for physicists. It commands overwhelming dominance over all other journals as a carrier of information between physicists of all lands.

2. American physics is the chief source of information, not only for other Americans but for the international community of physicists.

3. American workers in fields of applied physics (as typified by authors in JAP and IRE) find their literature needs overwhelmingly satisfied by American journals.

4. European physicists draw heavily on American literature. Their coupling to the Russian literature is not significantly greater than that of American physicists.

5. The Western world is virtually self-sufficient with regard to physics. The Russian cultural sphere on the other hand draws heavily on the West for its information.

A close comparison of the various tables suggests that these conclusions are valid even if we take into account the language barrier between East and West.

Flow of Information to an Applied Field

A significant special case of information flow and retrieval concerns communication across disciplinary boundaries. We suspect that a retrieval or communications scheme designed to process physics literature for the physicist is not the same as what is needed to process physics literature for the chemist, the engineer, or the biologist. A related problem, particularly important in the dynamics of applied research and development, is the flow of information from basic research scientists to production engineers. If the coupling is tight and information flows freely, one may expect a low lag time between basic discovery and application. A study of certain reference statistics indicates that definite patterns exist in this information circuit.

The June 1958 issue of the Proceedings of the IRE was chosen for study. This issue is a symposium of 22 papers (353 pages) devoted to transistor technology. Of the fifty authors, forty-one indicated affiliation with industry, eight with universities, and one with the government. Transistor technology is of great interest to industrial and defense workers and yet it is new enough to have rather simple and short routes to the underlying sciences. For these reasons it was thought that a detailed analysis of the transistor issue would be instructive. Table VII shows the distribution of references among journals.

Table VII

Journal Distribution of References in the Transistor Issue of IRE

Reference to	No. of References	% of Total
Physical Review	230	31.0
Proc. IRE	129	17.4
J. Applied Physics	69	9.3
Bell System Tech. Jnl.	52	7.0
British	48	6.5
German	61	8.2
Other Foreign	36	4.9
Russian, etc.	22	3.0
Miscel. Jnls. (American)	94	12.6

Tables VI and VII suggest that the transistor issue of the IRE shows the typical American pattern of bibliographic distribution and does not differ much from a random issue of the IRE. This invariance applies only if we consider the geographic distribution of references. Both cases follow the American pattern; over 75% of the references are to American and over 90% to Western journals. But when we analyze in detail the American journals for the two cases, an entirely different picture emerges. Table VIII is a break-down of the references to American journals in the two samples of the IRE.

Table VIII

Detailed Analysis of IRE References to American Journals

Articles in / References to	IRE	IRE Transistor Issue
Physical Review	7.3	25.2
JAP	2.7	7.5
Proc. IRE	23.0	14.1
Other American	43.0	26.3

Whereas the average issue of IRE refers to the Physical Review 7.3%, the special transistor issue has 25.2% of its references to the Physical Review. Thus we see that the bibliographic count is sensitive enough to measure the degree of coupling between science and technology.

The authors who contribute generally to the IRE have a lesser coupling to the Physical Review than those who are preoccupied with the new and rapidly developing field of transistors. This analysis of information coupling between science and technology can be continued another step. We see from Table VIII that 25% of the references in the transistor issue of the IRE are to authors of papers in the Physical Review. We now ask who are these authors? Do they differ as a class from the usual authors in the Physical Review? In other words, do the contributors to the IRE transistor issue make contact with a representative group of Physical Review authors, or is there some transitional group of physicists who serve as a bridge between the general population of physicists and the industrial group?

Table IX presents the institutional origins of authors who publish in the Physical Review, Journal Applied Physics, and IRE. In all cases column A refers to all authors in a random issue of the journal. Column B refers to those authors in the journal who were cited as references in the transistor issue of IRE.

Table IX

Institutional Distribution of Authors in Phys. Rev., JAP, and IRE

Authors in / Institutional Affiliation	Phys. Rev.		JAP		IRE	
	A	B	A	B	A	B
University	66	16.4	40.3	31.0	25.8	14.1
Industry	9	74.0	31.4	61.0	44.0	76.8
Government	13	6.3	17.5	5.4	10.0	0.6
Foreign	9	3.4	9.0	2.7	20.5	8.5
Others	3	-	1.8	-	-	-

The data show that ordinarily 66% of Physical Review authors have university connections and only 9% are from industry. The sub-group referred to by IRE authors more than reverses this picture; only 16.4% are university affiliated and 74% are from industry. Similar trends are apparent in the other columns of Table IX. On the basis of these limited data, it seems reasonable to assume that the coupling between basic science and its applied technology is tighter for the newer technologies and that the coupling is made through an intermediate group of scientists who form an intellectual bridge between the university and industrial community.

Flow of Information from the Past

At any given time scientists draw heavily on the accumulated experience of the past. Indeed, one of the greatest assets of the journal-article mode of communication is that it conserves the record of the past in an orderly and chronological manner. This coupling to the literature of the past was studied by plotting the number of references as a function of time into the past. Thus a distribution curve of references was obtained with age as the independent variable. Graphs 1, 2, 3, and 4 show such distributions for four journals. The integrated curves are shown in Figure 5. Although the curves are orderly, their statistical base is rather limited and one should be careful with conclusions. One may speculate that vigorous and fast-growing fields will show a sharp early rise and level off rather soon. Another hypothesis, however, may suggest that a sharp early rise indicates a superficial scholastic approach and that a more basic acquaintance with the literature of science would extend the curve farther into the past. Both hypotheses may be true, namely, workers in fast-growing, competitive fields may have no time to search the literature and thus confine their sources to current material. It is at any rate clear that this phenomenon has to be understood and absorbed into a serious retrieval scheme.

The Detailed Bibliographic Structure of a Single Journal

As a final example of bibliographic statistics, we mention a very detailed study that we made of the references in the Physical Review. The study was made for other purposes and involved a complete recording on IBM cards of all the references in 26 volumes of the Physical Review.* Some of the results are mentioned here because they relate to the subject of this paper. Excluding the unpublished and non-periodic literature, the authors in the volumes made 74,599 references to journal articles. Of this number 45,592 or 60% were to articles in the Physical Review. The next five most frequently used journals contributed another 13.8% of the references and the next twelve in order of frequency contributed 12.2%. Thus 18 journals accounted for 86% of all the references. The remaining 14% of the references were distributed among some 650 journals of which 240 were mentioned not more than once in all the 26 volumes and 420 were mentioned four times or less.

Another by-product of this study that is relevant to our discussion is the following. In spite of the strong definitive position of the Physical Review, it is nevertheless true that any given paper in the Physical Review has a very low probability of ever being used as a reference. Indeed, the largest single class of papers never appears in the reference literature at all. It is hard to assume that this large group of papers are never cited because they are worthless. Other reasons must be sought.

Discussion

The over-all impression left by the data may be summarized as follows:

a) If we consider the scientific paper as a message unit and the journal the message carrier, if we accept that inclusion of a paper in a published list of references indicates that the message found a relevant receiver, and if we regard the population of Physical Review authors as a representative group of physicists, then there is a massive asymmetry and an overwhelming inhomogeneity in the capacity of the many

* This work was done in collaboration with Mr. Frank Heart of the Lincoln Laboratory and will be published elsewhere.

hundreds of journals to serve as carriers of the scientific message. The imbalance may be due to language barriers, cultural and political isolation, reputation of the journal we have examined and its availability or to a combination of these. The inhomogeneity exists, whatever the reason, and in its most extreme form gives rise to the definitive journal, such as the Physical Review. The data raise important questions about the design philosophy of retrieval systems. In view of this inhomogeneity, should the retrieval process and the flow channels be the same for all carriers or should they take account of the carrier's capacity? If account is to be taken of the carrier's capacity, should the system be designed to further reinforce the strong and efficient carriers at the expense of the less efficient, thus reducing the noise, or should we on the contrary take the attitude that the efficient carriers need less attention than the weak, and therefore concentrate on the latter and raise their efficiency? Is the definitive journal a desirable phenomenon or does it in the long run inhibit the communication process?

How should we approach the complex problem of injecting the results of Russian research into the main stream of American physics. Such an injection is certainly desirable, but it is not clear that a massive translation program and wide distribution of the translated material is the best way of doing it. Another method might be to encourage a small number of practicing American physicists to learn the Russian language and depend on them as the instrument of injection. Considering that the money and effort available for this purpose are limited, one should not blandly accept either method.

A retrieval and communication system, unless its contribution is trivial, will have sufficient feedback to strengthen or weaken the various elements of the communication process that now exists. It is therefore important to understand these elements and to design our system with them in mind.

b) The previous section concerned communication within the relatively homogeneous group of physicists who are in the habit of publishing in the Physical Review. The problem of communication across field boundaries or between scientists and engineers is a somewhat different matter. The limited amount of data that we have collected must be considered as a sample that only indicates the complexity of the problem. It would seem that in this case there is no definitive journal. Furthermore, meaningful communication seems to involve a chain of intermediaries that form a bridge between pure science and applied technology. Should a system be designed to encourage traffic along this chain of bridges, or should it attempt to short circuit the chain? That this is not a purely academic problem is clear from the experience of the Chemical Abstracts, a major communicative link between physics and chemistry. The Chemical Abstracts attempts to short circuit any bridging mechanism and bring physics directly to each chemist by abstracting practically the entire physics literature. This, together with other examples of short circuitry, has so overloaded its own channel that Chemical Abstracts is becoming increasingly awkward and bulky. They could have chosen not to include the physics literature in its abstracts and to depend on the various hyphenated journals, such as the Jn. of Physical-Chemistry, Chemical-Physics, Colloid Chemistry, etc. etc., to act as injectors of physics into the main stream of the chemical literature. This would appear to loosen the coupling between physics and chemistry and increase the time necessary for information transfer. But the decrease in volume of traffic could well compensate for the looser coupling and produce a more efficient system. The phenomenon of coupling between fields is at any rate important enough to be considered in the design of a retrieval and communication system.

c) The flow of scientific information in time, past to present, or the useful half life of a scientific message unit is another important element of our problem. Our numbers indicate a useful half life of some five to ten years. A rigorous definition and measure of this phenomenon is not easy to come by. But it is obvious that no system of retrieval and communication can long survive without some method of purging its message population from time to time. This is not just a matter of eliminating poor material in favor of newer and better results. It is a curious fact that even the masterpieces of scientific literature will in time become worthless except for historical reasons. This is a basic difference between the scientific and belletristic literature. It is inconceivable for a serious student of English literature, for example, not to have read Shakespeare, Milton and Scott. A serious student of physics, on the other hand, can safely ignore the original writings of Newton, Faraday and Maxwell. The removal of a scientific paper from the retrieval system should not depend on a value judgment. The correct criterion should be based on the degree to which the paper's information has been metabolized into the flow stream of science. We could perhaps decide that once a paper has appeared in the citation literature a given number of times, it need no longer be carried as an independent message unit. This too needs more study.

At the other extreme we have the phenomenon of a large group of papers, perhaps the largest single group, that is never quoted in other people's bibliographies. If these papers do not enter the bibliographic literature within, say, five years of publication, they may become effectively lost to the literature. In view of the careful editing process, it is hard to believe that this large group of papers is useless or redundant. If they are useless or redundant, the publishers and editors could well afford to review their processing criteria. On the other hand, a continuing review of the citation literature over a traveling five-year period may reveal a group of papers that warrant more detailed channeling.

Concluding Remarks

Science and technology are approaching a crisis in communication. It would be a mistake to define this crisis entirely in terms of retrieval problems. Indeed, the break-down in communication between scientists must itself be evaluated in terms of the maturing sociology of science. Communication, after all, is not the only aspect of science that is in crisis. There are problems of technical manpower shortages, the increasing cost of scientific research, the growing imbalance between basic and applied research, and many others that relate to the emergence of science as a major instrument of national favor, stature and propaganda. But even if we confine our attention to the communication problems only, we must still remember that no single channel or mode is likely to solve all our needs. The components must be evaluated in terms of their contribution to the over-all system performance. Many systems problems must be studied if a working solution is ever to be achieved.

a) No system can be designed in an economic and social vacuum. Even the vital functions of national defense are subject to restraints. We must estimate the social and economic limits that will govern our system and optimize its function within these limits. How can limited resources best be apportioned between the various segments of the scientific community? What part of the resources shall be assigned to retrieval and other question-answer functions as opposed to directing the flow of information on the basis of generalized need-to-know criteria? What are the information needs within well-defined fields like physics as opposed to the flow of information across field boundaries?

b) If there is a critical failure of communication now, how will we know when improvement has taken place? What test procedures and criteria of performance can we use to evaluate the system? Unless a figure of merit can be assigned to the system as a whole, the contribution of any given component is always in doubt. One cannot test the performance of a system by measuring each component separately in terms of its own parameters. Experience with large multi-component systems indicates that one cannot generally arrive at a figure of merit analytically. It is usually necessary to build a model and evaluate new components in terms of their effect on the model.

c) To what extent should the system operate only when interrogated and to what extent should it operate on the initiation of its built-in logic?

d) Should the system be local, regional, or national? To what extent can communication media other than the journal article be exploited (radio, television, newspapers, remote printers, etc.)?

e) What is the probable cost of various systems, both initial capital investment and operating cost? Shall it be financially self-supporting or shall it be subsidized? Stability of financing is of particular significance because the system must have long-range continuity in order to be at all effective.

f) Finally, we wish to stress that the problems of science communication are not primarily equipment and hardware problems. Nor are they primarily problems in indexing, abstracting, or retrieval. The significant problems are in the area of systems design and systems logic. At this time there is no single organization whose professional competence and involvement embraces the entire spectrum of problems. Such an organization is needed if a reasonably successful solution is expected.

Reference

1. See for example the series of papers in "Area I - The Collected Papers of the International Conference on Scientific Information," Washington, D.C. (1958)

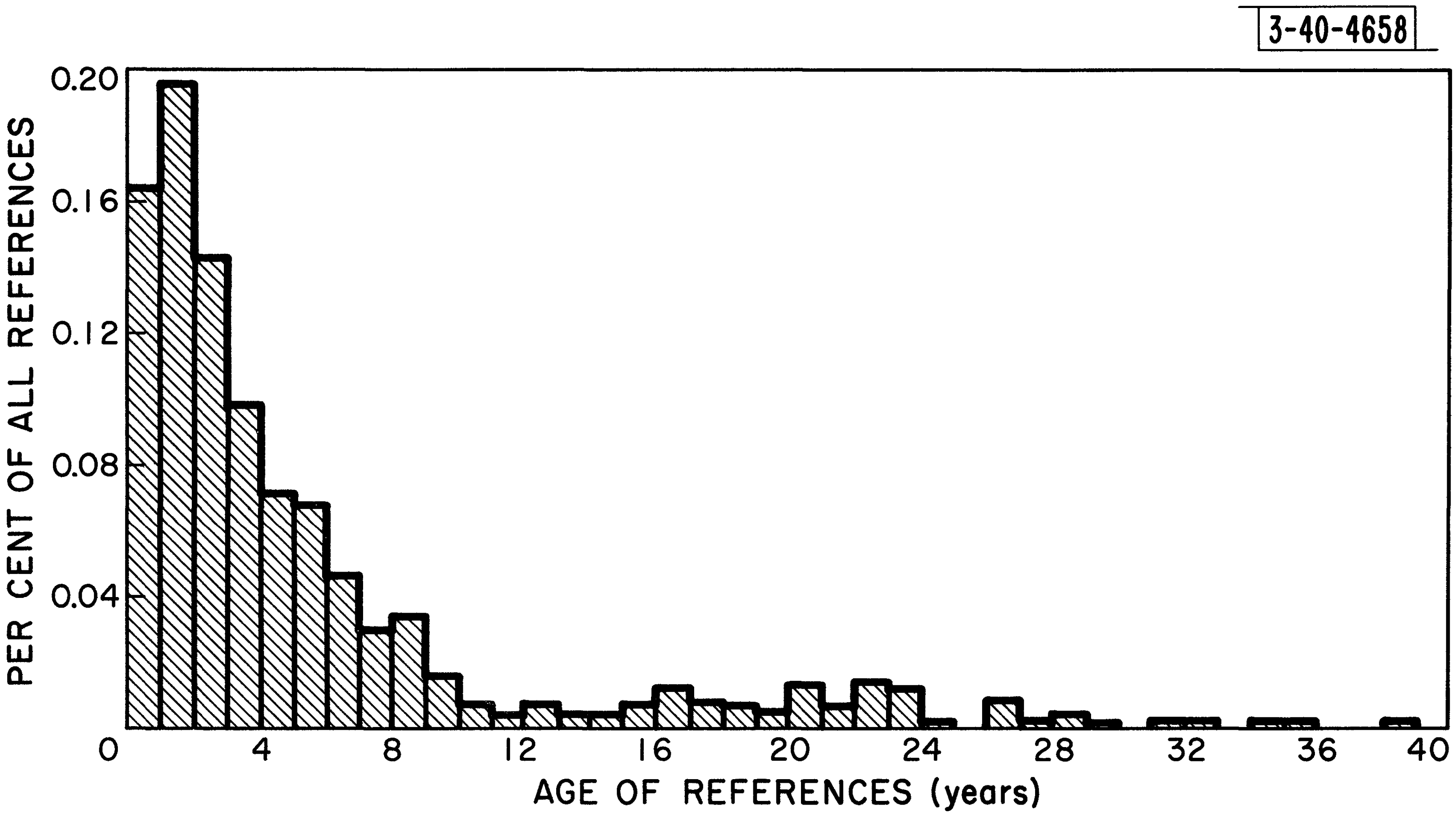


Figure 1 Distribution of References by Age
Physical Review 1957

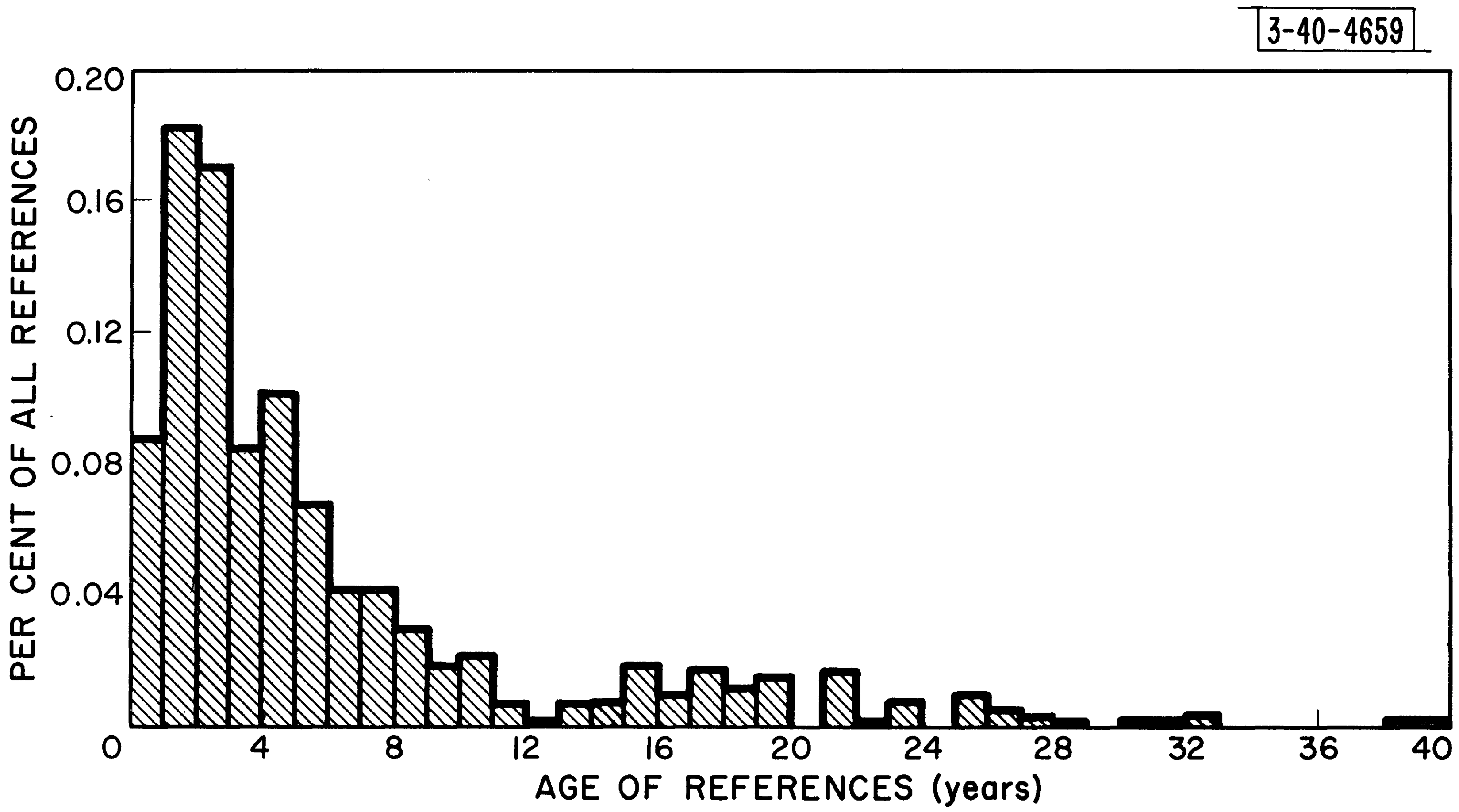


Figure 2 Distribution of References by Age
Russian JETP, February, March, April 1957

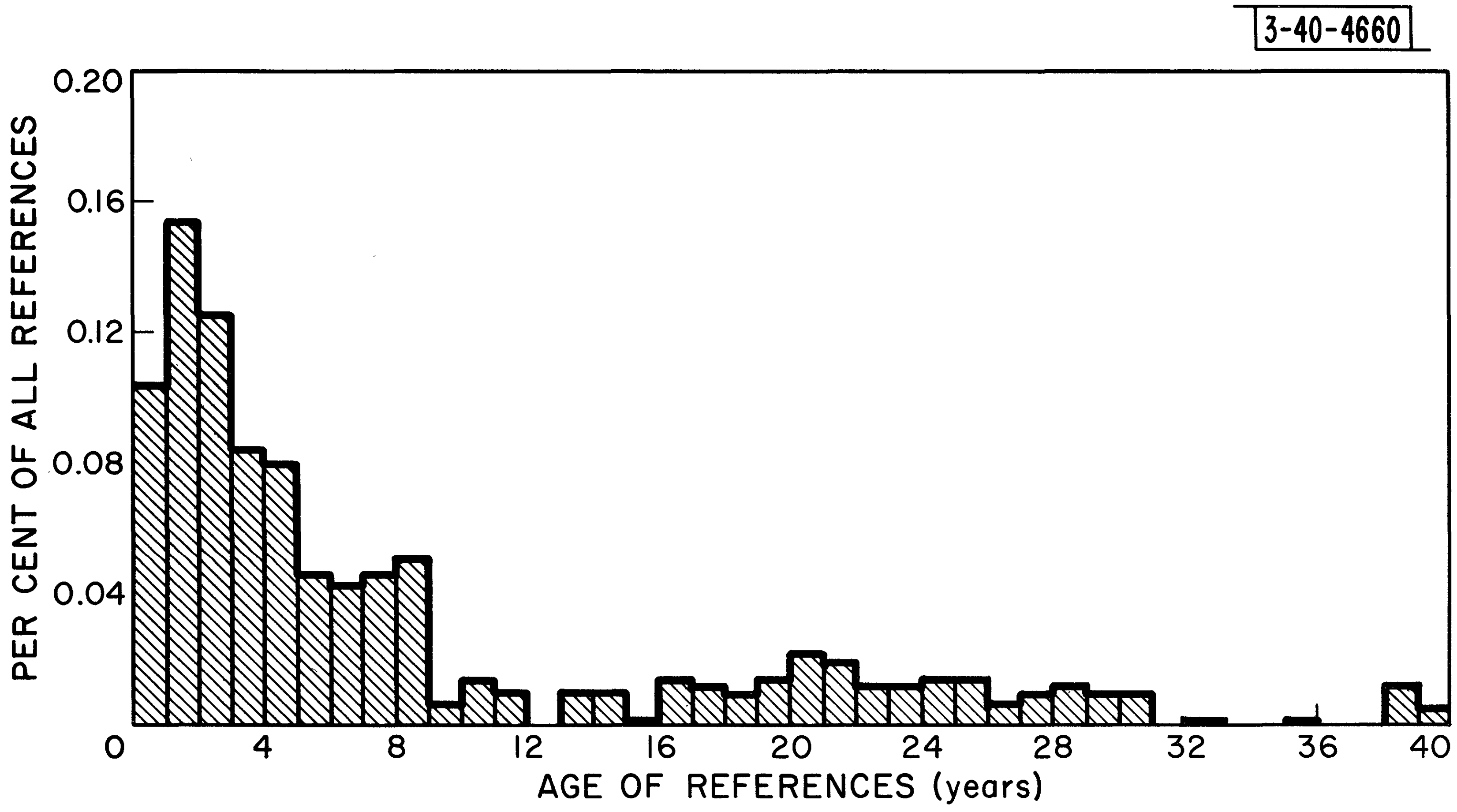


Figure 3 Distribution of References by Age
Physica 1957

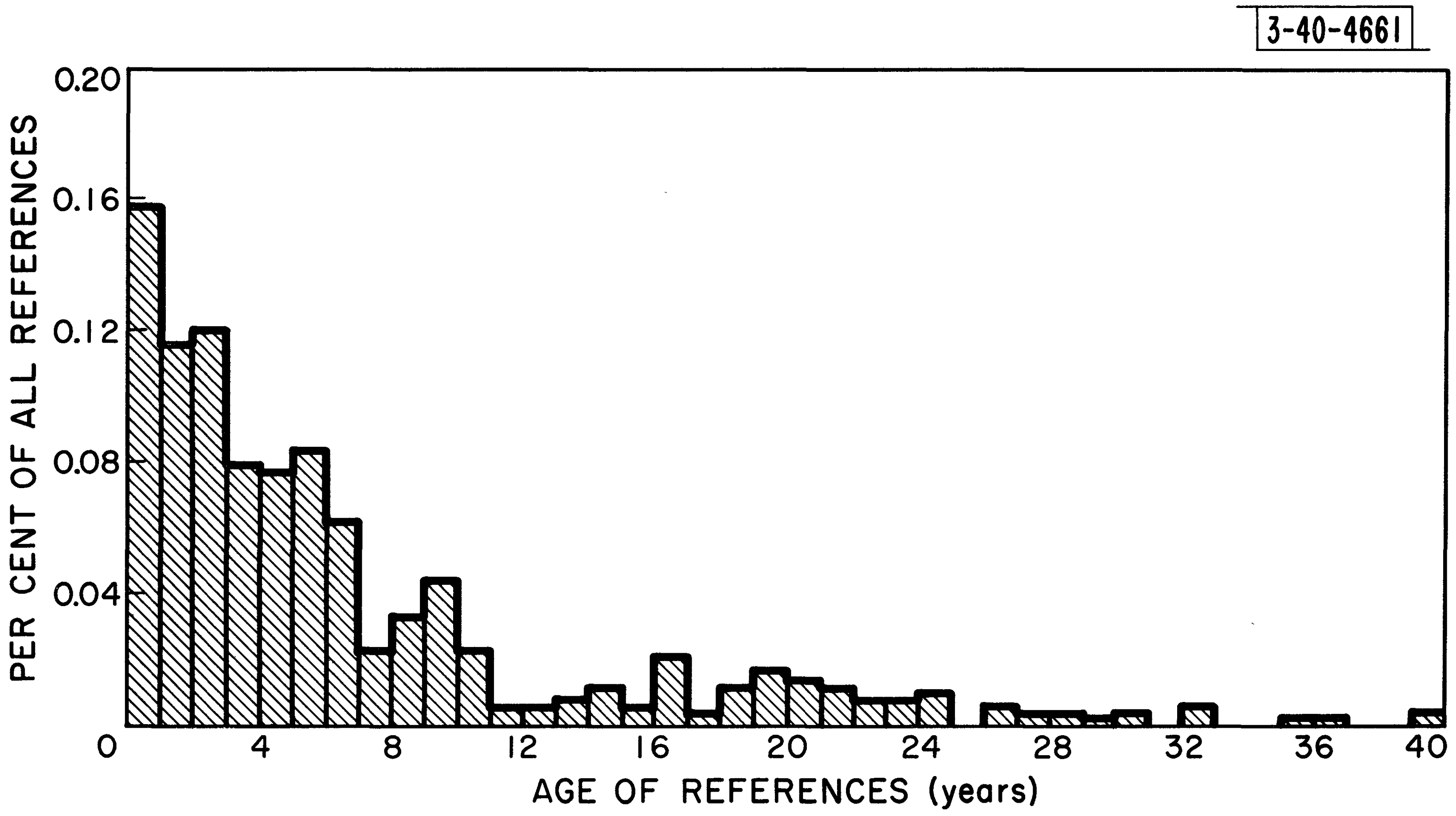


Figure 4 Distribution of References by Age
J. Acoust. Soc. Am., Jan., Feb., March 1957

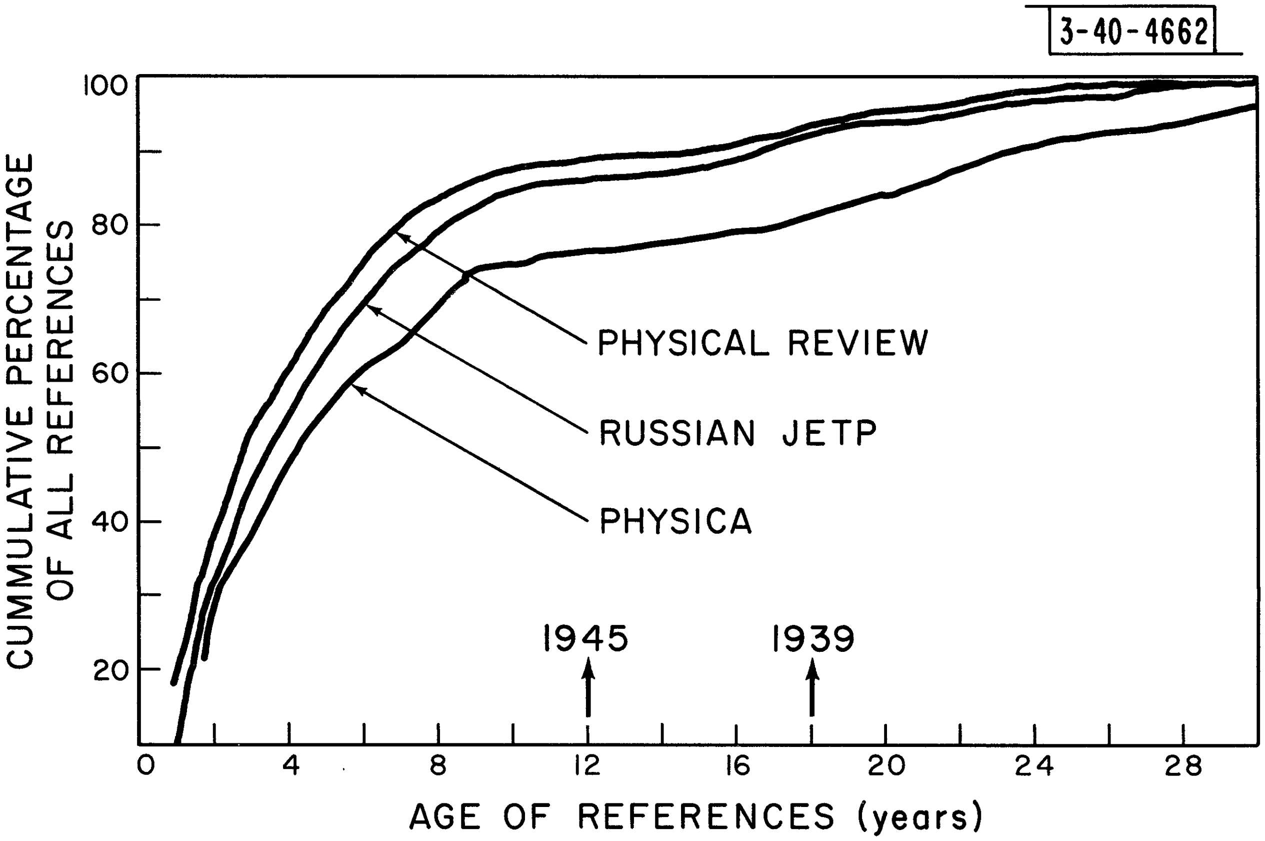


Figure 5 Cumulative Distribution of References by Age

A SCREENING METHOD FOR LARGE INFORMATION RETRIEVAL SYSTEMS

Robert T. Moore
Data Processing Systems Division
National Bureau of Standards
Washington, D.C.

Summary

This paper is addressed primarily to describing a method for reducing the excessive processing times sometimes encountered in large information retrieval systems. Two tools are suggested: (1) a screening system to allow multi-level processing of material and (2) pre-processing of the files to allow block rejection of documents not answering a retrieval request.

A screen using descriptors to constitute the first level of screening is described in detail, and a pre-processing technique, based upon clustering of descriptors, is developed. Two methods of implementation on high-speed data processors are described based on serial access type storage and random access "dynamic storage," respectively.

This screening technique is compared for efficiency against two other familiar methods of employing descriptors: the ordinary serial descriptor list method and the inverted file technique.

Introduction

The primary purpose of this paper is to describe a method of constructing an initial screening stage for a multiple-component information retrieval system. The screening system (or sub-system) utilizes document descriptions comprised of independent single-word index terms, hereafter referred to as "descriptors" which are thus to be interpreted in their most restricted connotation, and applies a method of file organization and compression which has several special advantages. Two different methods of implementation will be discussed, the choice of method being dependent upon the type of data processing equipment to be used.

The designer of a large retrieval system is faced with two separate but related problems. He must minimize the expensive machine time required to process a request for information. It is also desirable to minimize the total storage, both internal and peripheral, required for the index file. The system described here represents an attack primarily upon the processing time but some of the ideas introduced show promise of reducing the file size as well.

In the process of discussing this specific screening system, several ideas of greater generality are utilized, among them (1) the "screening approach", as applied to storage and retrieval systems, and (2) the redundancy encountered among "descriptions" of different documents in the file, which may be applied to compressing and structuring a large file of retrieval information. Each of these ideas is abstracted and discussed in a more general context before being applied to the specific system under consideration here.

Screening and the Use of Descriptors

Advantages of the Screening Approach

It is useful to distinguish between two alternative approaches to the problems of information storage and retrieval. The importance of this distinction, it is hoped, will be adequately illustrated in the discussion to follow.

One approach is to look upon an information retrieval system as a "mechanized librarian". The user explains his problem or question to the librarian, and is given suggestions as to where in the library he may find the answer. If his question turns out to have elicited no helpful suggestions, he rephrases it and tries again. The function of the librarian (mechanical or human) is to compare the request with "stored" information about the library and, after considering various "aspects" of the question, to make determinations of "relevance", thus directing the user to the right documents. This might be described as the searching approach, since it involves the librarian's actively seeking the relevant material.

Alternatively, we may design a retrieval system which behaves in a manner that would be considered obtuse, at best, in a human librarian. It can single out all those references which could not possibly be of any use and remove these from consideration. It serves as an information garbage disposal unit, which discards the definitely useless and presents the remainder to the user. Since this represents a screening out of the undesirable documents (to change metaphors), it may be called a screening approach.

Of course, for perfect systems this distinction is uninteresting. Any document that "passes through a perfect screen" will be "found by a perfect search" and vice versa. What is of interest is the way in which an imperfect searching system fails, as contrasted with the comparable problems in an imperfect screening system. An adequate searching system will only rarely "find" documents which are irrelevant, but it may very well fail to locate documents which a human reader of all the documents would classify as pertinent to a request. An imperfect screen, on the other hand, will not block good documents nearly as often as it will allow documents to pass which a human reader would know were irrelevant. (These may be considered as the defining properties of the two types of systems. However, they also reflect two different philosophies of system design.)

In all of the discussion to follow, we presume that the indexing is such that the systems discussed are imperfect, since no one seems to have produced a system which cannot be confounded by the vagaries of "semantics". However, the problem of human or mechanical error is not to be considered.

In the National Bureau of Standards-Patent Office HAYSTAQ project[1], we are finding it necessary to adopt the screening approach as a supplement to the searching approach in order to solve the particular set of problems with which we are faced. In the HAYSTAQ program, our more elaborate and sophisticated programs (topological comparison of chemical structure diagrams, comparison of logical expressions, etc.) are too detailed and time consuming to be run against every document in files of the size we will eventually have to handle. Thus the needs of the HAYSTAQ system point directly to the desirability of a simple economical screen to reduce the volume of input to the more elaborate routines. Such a screen would also be a logical starting point for the construction of new systems. The main subject of discussion in the remainder of this paper is a system of that type.

Descriptors for Screening

Although mutually independent descriptors or index terms have been widely used since the early years of information retrieval research, more recently many workers in the field have begun to question their value for projects requiring indexing in depth.[2] They argue that scientific discourse is too complex to allow adequate indexing by means of unconnected one-word comments on a document.[3] While it is demonstrable that simple descriptor methods are inadequate as a single solution to problems of deep and exhaustive indexing, this by no means implies that they should be rejected entirely. In fact, their very simplicity adapts them very well for use in initial screening systems.

The key to setting up a nearly fail-safe descriptor screen naturally lies in the type of vocabulary selected. The vocabulary should be a list of objects, processes and relations discussed in documents in the file. A vocabulary which classifies documents ("physics", "mechanical translation", etc.) or describes them in general terms cannot be fail-safe because of the "peripheral" vagueness in the meaning of such terms. An index using these terms will also tend to become obsolete as science and technology progress. Ideas and techniques developed in one field often find application in seemingly unrelated fields.

One can easily get a list of candidates for the vocabulary by listing <u>all</u> objects, processes, and relations that occur in a reasonable-sized random sample of the documents in the library to be indexed. The problem is then to choose that small set of terms which has the greatest "power of discrimination" among documents in the library. In calculating power of discrimination, one must also consider how often terms will be used in framing requests. (See discussion of measure of merit in the Appendix for an example.) It is not our purpose to discuss solutions to this difficult problem. Suffice it to say that it is closely related to the problem of character recognition: What is the cheapest way to distinguish between centered binary images of the letters of the alphabet? What bits in the images carry the most information?

A descriptor set for a document will then be a simple list of the descriptors applying to the document (objects, processes and relations discussed in the document) chosen from the selected vocabulary. Then when a given request is being processed, a document will be screened out, if and only if there is one or more descriptors in the request that is not in the list applying to that document. If the screen is to function in a fail-safe manner, the user must keep this fact in mind when he sets up the descriptor list of each request.

File Preparations

General Method

Rather than use a file which is encoded in the language used by the human document analysts and arranged in the order in which the documents are prepared for machine storage, it is altogether reasonable to transform it into a "pre-screened" or "pre-searched" file. Such a file would be organized so that documents could be processed partially in parallel against a request. Furthermore, many of the manipulations and housekeeping operations which are independent of the specific content of a request can be performed during the file preparation process in advance of processing of requests. (An example is the "inverted file" investigated by Nolan and Firth[4] and used in several Patent Office experiments.[5]) It should be noted that such prepared files are of great utility in either searching or screening systems, although our attention here will be confined solely to the latter application.

When an information storage and retrieval operation is to be carried out, one chooses an appropriate symbolic language, which can be manipulated by computer (English text is one possible example). Document descriptions are then encoded in this language. In many cases there will exist several equivalent symbol sequences which could describe a given document. ("Equivalent" should be taken to mean that there are rules within the formal language which allow one to transform one expression into another and the converse.)

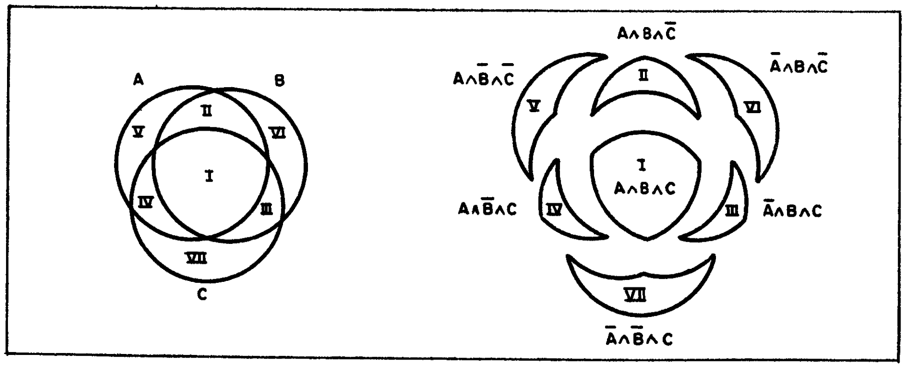
From this point on, the symbolic language will be regarded as fixed. The sequence of symbols representing each document will be one of a class of equivalent expressions, i.e., only transformations of the symbol string from its original form to an equivalent form are allowed. One might think of the transformation $P \vee Q \longrightarrow Q \vee P$ in the propositional calculus as an example of this type of transformation. In this section, no further reference will be made to the meaning of the symbol strings involved; only syntactic or formal properties will be employed.

We then consider possible redundancy within a file. .It is difficult to give an exact definition of what is meant by "redundancy" in so general a context. Reference to one of the specific symbolic languages is needed in order to be precise. Basically what is intended is that if a certain configuration of specific symbols occurs repeatedly in many document descriptions in the file, it carries less information per symbol (in the information theoretic sense) than does a comparable configuration which occurs only rarely.

If a "metasymbol" or "molecular symbol" were assigned to each commonly occurring configuration (which is composed of the "object" or "atomic" symbols in the original language), the file could be in some sense recoded in terms of these metasymbols. The recoded file would potentially require substantially fewer bits to be stored in memory and might be organized to place documents with metasymbols in common near one another to facilitate group inspection.

It is helpful to visualize a document description as a set or as a circle on a Venn diagram. The metasymbols are applied to intersections or "overlaps" of these sets. (See Figure 1.) In at least some of the possible symbolic languages (descriptors, propositional calculus, predicate calculus, and topological structure codes), it is possible to develop this line of reasoning beyond the heuristic level by imposing an appropriate Boolean algebra on the symbol strings. The use of Boolean algebra provides no startingly new information about the metasymbol techniques, but it serves as a convenient and compact language in which to discuss the topic. It permits us to abstract only those aspects of the situation which are useful, so that we do not become bogged down in irrelevant detail.

In files in which there is an equivalence class of expressions for each document in the library, the size and frequency of occurrence of overlaps are strongly dependent upon which representation is chosen for each document. It then becomes necessary to discover an algorithm for finding an optimal canonical representation for the file, i.e., one which maximizes the overlaps and selects a unique representative for each document. Fortunately, in the descriptor case there exists a unique representation for


A
B
C
I
II
III
IV
V
VI
VII
$A \wedge B \wedge \bar{C}$
$A \wedge \bar{B} \wedge \bar{C}$
$\bar{A} \wedge B \wedge \bar{C}$
$A \wedge B \wedge C$
$A \wedge \bar{B} \wedge C$
$\bar{A} \wedge B \wedge C$
$\bar{A} \wedge \bar{B} \wedge C$

Regions I - VII correspond to metasymbols

Figure 1

each document, so the problem is entirely avoided here.

Utility of Metasymbols in Screening

A set of metasymbols for an index file (a file of symbol strings for a library) is of some intrinsic interest, since it is likely to correspond in many cases to clusters of ideas that are discussed together in the literature. Such "co-occurrence" phenomena may lead to some interesting information about the structure and interrelationships of various scientific disciplines represented in the basic library. Such possibilities rest upon a number of debatable epistemological assumptions about the connection between the content of a library and the syntactic relations between expressions used to encode that content. The teeth of any such arguments must be sharpened on a plentiful supply of data before they can be expected to cut into the problems of information retrieval.

Fortunately, the "conceptual significance" of such metasymbols need not concern us here. So long as they exist at all (which is tantamount to saying "the file is not a completely random set of symbol strings") we can put them to work in screening our files. Their utility lies in the possibility of "block screening". Every metasymbol represents the common overlap of a set of documents and may be said to apply to each document in that set. Suppose that when a request (sequence of symbols) is directed to the file, comparison of this request symbol string indicates that <u>no document to which the metasymbol applies</u> could satisfy the request. Then every document to which the metasymbol applies can be rejected outright. No further information about these documents, e.g., which other metasymbols apply, what are their identification numbers, etc. need be examined. In other words, documents can be rejected in large blocks if metasymbols are used.

Hence, after discovering a fruitful way of recoding a file in terms of metasymbols, we concern ourselves with finding methods of exploiting this "block rejection" in as efficient a manner as possible.

In fact, one of the best measures of the value of meta-recoding a file will lie precisely in the degree to which it can be applied to the idea of block rejection of irrelevant documents. If a document will eventually be rejected in response to a given request, how early in the process (on the average) can it be rejected? Is rejection information used as soon as it is obtained, or must rejected documents be carried as excess baggage for a time? If a document is to be accepted, how much extra manipulation is necessitated by the recoding? How much extra "bookkeeping" information is introduced by the recoding?

Questions such as these must be answered before any proposed system can be evaluated. We will now describe a recoded file using descriptors and see what kind of information such questions elicit in this particular case.

A File Structure for Descriptor Screens

Since our discussion is limited to the problem of designing a file structure for use in descriptor screening, we can develop the set analogy explicitly and demonstrate its utility, as well as give a concrete example of a symbolic language. It will become clear that the simplicity of descriptors and the application of the screening approach both contribute to the workability of the structure.

Initially, the documents are indexed according to which of the N descriptors apply to them. However, it is more convenient for screening to represent them in terms of their "rejectors" or descriptors-that-do-not-apply. Also, a binary number notation is often used in connection with descriptor methods.

The N-word vocabulary is then organized in some convenient order, e.g., alphabetical, and an N-digit binary number or Boolean N-vector is constructed for each document by putting a 1 in the <u>i</u>th position if the <u>i</u>th descriptor applies or a 0 in that position if it does not. Note that this is the usual "fixed field" method of handling descriptors, particularly in punched card applications. This technique is of great use here, although the logical complement of the usual Boolean vector will be used (this is the vector for the rejector list of the document). This complemented number will be called the <u>rejector vector</u> for the document. Since the set-theory operations of intersection $(A \wedge B)$, union $(A \vee B)$, and complementation $(\overline{A})$ correspond directly to the Boolean or logical "and", "or", and "complement" as defined on the rejector vectors, any manipulation of rejector vectors can be pictured in a suitable Venn diagram and vice versa.

Using these tools, we are equipped to talk about the appropriate metasymbols. Giving up the logician's prefix "meta" in favor of the more suggestive chemist's term "molecular", henceforth, in this paper, we shall call the metasymbols "molecular rejectors". (Note that these behave more like <u>radicals</u> than <u>molecules,</u> so the chemical analogy cannot be pushed too far.) A molecular rejector will be the intersection or "and" of some collection of rejector vectors, that is, the set of 1-bits common to <u>all</u> the rejector vectors in that collection.

Our purpose here is to describe an algorithm for selecting a set of molecular rejectors for the file. In the interest of efficiency as previously discussed, this particular scheme defines molecular rejectors which in many cases "make sense" only for a subsection of the total file. The molecular vocabulary then has a highly variegated "local" aspect, and the different molecular rejectors are not necessarily disjoint. Nevertheless each document will have a unique molecular characterization.

The guiding principle is that of maximizing the average size of blocks rejected and of carrying out this rejection as early and as completely as possible. It is possible to develop a rigorous theory of the redundancy of a molecular rejector S, but this is an involved process. Such an analysis is important in choosing the method of storing atomic definitions of molecular rejectors that minimizes storage requirements. This will not be taken up here, however.

The quantity of interest is the average number of documents in a given subfile G rejected by a given molecular rejector--a measure of merit. This measure $\mu(S)$ will be the product of the number of documents, t, which would be rejected by S, t(S,G), and the probability P(S) that the request will overlap with S$[\mu(S) = t(S,G)P(S)]$.See Appendix for a discussion of possible approximate functions for $\mu(S)$.

To find a structure for the file F we proceed as follows:

1. Generate all 2^N-1 possible molecular rejectors for the file F.(Ingenuity may make it possible to reduce significantly the number to be generated.)

2. By appropriate manipulations, compute $\mu(S)$ for each molecular rejector and select the one of highest μ as S_1. If there is more than one S of the same maximum μ, life becomes complicated. We could pick one of these at random for S_1. (See the Appendix for additional discussion.)

3. Decompose the file F into two disjoint subfiles, F_1 and F_2':

   a. F_1 contains all documents which S_1 rejects -- all of the $t(S_1F)$ documents having at least the atomic rejectors, or 1-bits, of S_1 in their rejector vectors.

   b. F_2' contains all documents of F not in F_1 -- those which lack at least one 1-bit of S_1.

4. Repeat the procedure of steps 1 through 3 for F_2' to generate S_2. (F might be called F_1' for consistent notation.) We then have F_2 which S_2 rejects and F_3' which it does not. This procedure may be repeated, always operating on a "primed" file to produce an "unprimed" file with the same subscript, a molecular rejector with the same subscript, and a new smaller "primed" file with its rightmost subscript one greater than that of the subfile from which it is derived. It is possible, when attempting to apply this process, that the primed subfile will turn out to contain a number of "rugged individualist" rejector vectors which do not group well with one another, i.e., all $\mu(S)$ in this subfile are very small, less than 1, for example. In this case, each rejector vector in the subfile is given its own molecular rejector vector, which completes its definition and produces no new primed <u>or</u> unprimed subfile.

5. Apply the following procedure for F_1 (note that it is slightly different than that applied for F_2'): Since all F_1 rejector vectors have S_1 in common, it is necessary to disregard these bits in any further processing. This is accomplished by "subtracting" S_1 from all the rejector vectors, A_i, of F_1 $(A_i \wedge S_1)$ before proceeding. Then steps 1 through 3 may be applied to the modified F_1 generating S_{11}, F_{11} and F_{12}'. In general, this process, when applied to an unprimed subfile, generates a new smaller unprimed subfile with the new subscript q to the right of those subscripts applying to the old file $(F_{ij\ldots p} \longrightarrow F_{ij\ldots pq})$, a new molecular rejector with the same subscript $(S_{ij\ldots pq})$ and a primed file with its rightmost subscript increased by 1 $(F'_{ij\ldots(p+1)})$.

In some cases of application of step 5, the subtraction process will leave some of the rejector vectors in the modified file (modified F_1 in the case explicitly considered) completely empty of 1-bits. This means that these have been completely characterized by the existing list of molecular rejectors, so the corresponding document numbers, with complete molecular rejector list, can be set aside for later insertion into some storage unit. The process is then continued on that part of the modified subfile containing non-empty rejector vectors.

Another possibility is that the newly generated subfile contains either only one rejector vector or a set of identical vectors (applying, of course, to different documents). In both cases, the remaining rejector vectors are assigned a single final molecular rejector and classified completely. Hence no unprimed file is generated in this case.

The procedure just outlined, when developed in detail, leads to a complete characterization of the rejector vector of every document in terms of molecular rejectors. If A_1 is the original rejector vector and S_j, S_{jk}, ... $S_{jk\ldots p}$ are the molecular rejectors assigned to it, then $A_i = S_j \vee S_{jk} \vee \ldots \vee S_{jk\ldots p}$. Figure 2 is a rough flow chart for one possible algorithm for accomplishing this, and Figure 3 sketches the "generation tree" for some file.

At this point, one might set up a dictionary of a sort, defining each molecular rejector in terms of the 1-bits it contains, and then store the file described in terms of these molecular rejectors. Actually the subscripts of the rejector with the most subscripts which applies to a document suffices to specify all its molecular rejectors, as study will indicate. Such a procedure would lead to no particular advantage, however. The full potentialities of block rejection can only be realized if a somewhat more elaborate storage method is used. Two such methods will be described in the next section.

The File Growth Problem

In the majority of applications, the library will not be a static collection of documents, but one which is continually growing. Hence it is necessary to examine the possibility of incorporating the rejector vectors for new documents into the structured index file. The file described above can accommodate considerable expansion easily, particularly if new material has the same basic statistical properties as the original file, i.e., all of the $t(S,G)$ in the augmented file are nearly proportional to those of the original.

It is not difficult to insert a new document into an organized file of the sort described. What is required is an algorithm which will give a unique molecular characterization of the rejector vector of the new document. If A_{M+1} is the new rejector vector, A_{M+1} is compared with $S_1, S_2 ...$ until the S_j of lowest index which applied to it ($A_{M+1} \wedge S_j = S_j$) is located. If no such S_j exists, all of A_{M+1} becomes a new single index molecular rejector. If an S_j does exist, then it is applied to the document and subtracted from the rejector vector. The remaining rejector vector is then compared with S_{j1}, S_{j2}, ... until again either one of lowest right index is found which applies or it is found that none applies. Both cases are treated substantially as before, and in general the process is repeated until, in one manner or another, the new rejector vector is fully described by a set of molecular rejectors (at most one of these will be a newly generated one). In most cases a reasonable number of pre-existing molecular rejectors will apply, and only a few atomic rejectors will remain to be lumped into a new molecular rejector.

Clearly, however, it will eventually become profitable to reprocess the file and generate a complete new set of molecular rejectors, because new accessions will have rendered the old statistics invalid, and a less efficient file will have been built up. The file structuring method outlined here does not really come into its own until the file is large enough so that the percentage increase (per year, say) is small, and the statistics are really representative of the type of information being inserted. (In scientific and technical literature, progress and changes in the areas attracting interest will tend to cause systematic drifts in file statistics.)

Storage and Screening Schemata

Conventional Techniques

The usual method of approaching the storage and search problem consists of storing the index file on some permanent medium (punched cards, tapes, drums, discs, etc.), often storing auxiliary tables such as dictionaries, and providing an internally stored "search program" for a high-speed digital data processor. In our application, this program takes requests as input and delivers as output the identification numbers of those documents which pass screening.

The storage-and-screening method to be described here is especially well adapted for tape storage or other storage in which serial access is necessary. It minimizes storage requirements by using tape position to store much of the information. All molecular rejectors are defined exactly once on the tape, and every document identification number is listed exactly once. Relative position on the tape determines which molecular rejectors reject which documents. In the process, various marker symbols are used. These may, in practice, be fields of a few bits in words used primarily for molecular rejector definition, but in the discussion to follow they are regarded as independent entities.

In this storage, one begins with the marker symbol M_0 and then gives the atomic definition of the molecular rejector S_1. It is helpful to visualize this as the list of atomic rejectors for S_1, stored as a rejector vector. However, in general this would be a comparatively inefficient use of storage space, since S_1 itself is composed primarily of 0-bits. The S_1 definition is followed by M_1, S_{11}, M_2, S_{111}, M_3, etc., until the complete molecular sequence for one or more documents has been completely defined. Then a T (terminal) symbol is written, followed by a list of the identification number(s) of the(se) document(s).

At this point two different situations may arise:

1. All of the molecular rejectors which reject the first document(s) (suppose there are k of these) may reject more documents, but not constitute a complete characterization of the rejector vectors for these documents. In this case, M_k is written after the identification numbers of the documents, followed by $S_{11...11}$ (k + 1 "ones" in subscript), M_{k+1}, etc. until another document is completely described.

2. The molecular rejector $S_{11...1}$ (k "ones" in subscript) may apply only to the first document rejector vector, but the other k-1 molecular rejectors apply to more documents

(in a more general case, this will be $k-p$, $1 \leq p \leq k-1$). Then M_{k-1} (M_{k-p}) is written after the document identification numbers followed by $S_{11\ldots12}$ (k subscripts in all), M_k, and so on, until another document rejector vector is completely described.

This will suffice to suggest the general method of listing document numbers and defining molecular rejectors. This method is iterative. Figure 4 gives a diagrammatic example of a possible tape configuration.

The file is screened in a particularly simple way. One takes the request descriptor vector and begins at M_0 on the tape. The request is compared with each molecular rejector in turn until one of the following conditions occurs: (1) a T symbol, along with a list of one or more identification numbers, is reached; or (2) following M_i, the molecular rejector definition contains 1-bits where the request also contains 1-bits.

In the first case, none of the molecular rejectors characterizing the rejector vector of the listed documents has any bits in common with the request; hence the documents in question must possess at least those descriptors given in the request, and the documents have passed screening. Therefore, the identification material may either be printed out or passed on to a more powerful routine which calls in a more detailed request and compares it with a more detailed index file.

In the second case, the "bit overlap" between request and molecular rejector means that none of the documents to which the rejector applies can possibly satisfy the request, since they all lack at least one descriptor that is asked for in the request. Therefore, all further molecular rejectors pertinent to these documents, as well as their identification numbers, can be ignored, and the tape can be advanced immediately to the next occurrence of an M_j with $j < i$, and screening can be begun again at this point. (See Figure 4 for an example of this process.)

Clearly this procedure avoids the necessity of examining many of the molecular descriptors in the file. The full rationale for the file structuring procedure now becomes clear, since it assigns molecular rejectors in such a way as to maximize the number of documents screened out in a single operation and also to increase the probability that rejector-request overlaps will occur early in the screening process.

Serial storage has the one drawback that it is costly to add new rejector vectors (with document identification numbers), since these will in general belong somewhere in the middle of the file, and recopying a part of the file tape will be necessary to accomplish this. It is also input-output limited, since there is a bare minimum of "computation" to be done. The best use of this screen would be as the input-controlling subroutine of a larger routine (as suggested above), preferably one with much greater internal processing. (It may be used in just this way in connection with the HAYSTAQ chemical structure search program.)

Random Access Integrated Program-File

If very large random-access addressable storage is available constituting, for example, more than half of the total required storage, another method becomes feasible. The screening program and the file may be integrated into one unit, through which, in a manner of speaking, control "propagates" until it reaches an output point where print-out occurs.

In this method, the same basic organization is used as was discussed under Conventional Techniques, but the molecular rejector and its marker are replaced by a subroutine which examines the request and, depending upon what bits occur in the request, passes control directly on to the next subroutine or diverts it to the subroutine which replaces the Mj ($0 \leq j \leq i$). These subroutines might actually have the rejector vectors of their molecular rejectors stored with them, which are compared with the request, but other more efficient alternatives are available. (See Figure 5 for a sample flow chart.)

This sort of file-program has the probable drawback of requiring more storage than the procedure outlined earlier (experimental data are needed here), but it is faster in that it avoids any necessity of passing over rejected material. It proceeds directly from one comparison to the next, with great consequent gains in speed.

A compromise between these two methods would be to keep part of the screening routine distinct from the file, replacing the marker M_i with an instruction which loads an appropriate counter with the address of the molecular rejector which is associated with M_i in the prior procedure. This counter would control the "work area" of the routine and would normally advance by one, unless modified by an "M instruction".

Such methods could be most efficiently applied in systems with very limited control units, having only a few indexing, logical, and conditional transfer instructions in their repertory but very large address fields, and a very large bank of addressable memory, for example, disc storage. The random-access feature would make it quite simple to enter new data into the file, since appropriate adjustment of transfer (or indexing) instructions can allow "logically contiguous" material to be stored in widely separated sections of storage. (The author would like to acknowledge the relationship between this and the comparable techniques employed in the various list processing routines.[8] It may be possible to use list processing techniques directly here without loss of efficiency, but this seems unlikely, at least as long as elaborate compilers and interpretative routines are necessary to the use of list processing.)

Evaluation: Comparison with Ordinary and Inverted Files

The Files from a List Viewpoint

The best way to evaluate the worth of the new file structure described here is to compare it with the more familiar ordinary and inverted files. Then the questions posed in the section on Utility of Metasymbols in Screening can be answered in a comparative way, and the new file put into some sort of perspective. In order to compare the files and the methods by which requests are handled, we need to describe them in a common language. The terminology of list processing (used loosely) is well adapted to this description. We will find it convenient to describe some files in terms of descriptors, others in terms of rejectors.

The ordinary file. In this type of file, the primary list is the list of M document numbers. Each document number then serves as the point of attachment for an index sublist. These appended sublists may be either descriptor lists or rejector lists, although the latter may be cheaper to process. In the processing, a request descriptor sublist is compared with the sublists for each document in turn to see whether all requested descriptors are present (or that no request descriptor occurs as a rejector), and the document number is added to an "accepted" list if it passes.

The inverted file. A file is normally inverted with respect to its descriptors. Then the primary list is made up of the N descriptors. To each descriptor is attached a sublist of those documents to which it applies. A request descriptor list can be processed by use of a "candidate list". This list is initially the document number sublist for the first descriptor that appears in the request. Each additional request descriptor is processed by obtaining from the file the document number sublist attached to it and comparing this sublist with the candidate list. All candidate numbers which do not appear on the descriptor sublist are removed. That is, only those that appear on both the descriptor sublist and the candidate list are retained (thus taking the "and" or intersection of the two lists). If the file were to be inverted with respect to its rejectors, one would combine the sublists for all those rejectors appearing as descriptors in the request. The result would be a list of rejected documents, which would have to be "subtracted" from the list of all documents.

The new file. In the case of the file described in earlier sections of this paper, a more elaborate and unconventional list structure is required. The molecular symbols $S_{ij\ldots p}$ participate in the structure, as well as document numbers and rejectors. The primary list is a list of single-subscript molecular symbols. Each has attached to it two sublists:

1. A rejector sublist which defines the molecular rejector in terms of its atomic rejectors;

2. A sublist of all two-subscript molecular symbols which have the subscript of their primary molecular symbol as the first subscript. In the case of a "rugged individualist" rejector vector which has been made a single index molecular rejector, this sublist will contain the document number or numbers which the rejector vector would single out for rejection.

In turn, each two-index molecular symbol will have attached its two sublists, and so on, creating a highly branched tree of lists. The logic of the screening procedure for this set of lists is essentially that previously described under Storage and Screening Schemata. We observe that, in practice, the "list" method would be impractical here, since the molecular symbols are excess baggage.

Before leaving this topic, it is interesting to examine the relationship of the new type file structure to the ordinary and inverted file types. The ordinary type, it turns out, is nearly a special case of the new type and the inverted file is a close relative of another special case.

If every rejector vector of the new file is treated as a "rugged individualist", then there are no molecular symbols with more than one subscript. Hence, the file becomes a single list of molecular symbols, each "defined" by a rejector list and paired with one (or more) document numbers. With the exception of the superfluous molecular symbols, this is a refined version of the ordinary file. However, unless all M documents have different rejector vectors (or lists), there will not be M entries in this primary list, since all documents with any given vector are combined in one entry. This is clearly a sensible innovation in any case.

On the other hand, if each of the molecular rejectors contains exactly one atomic rejector, the file structure that results becomes a multi-level file inverted with respect to rejectors. If the file were inverted with respect to rejectors, and each of the document sublists inverted again with respect to the remaining rejectors, and so on repeatedly, a closely related file structure would be obtained. In this case, however, every document appears b times in the file, if it has b rejectors. In the one-rejector case it appears only once. Basically, the relational diagram of the multiply-inverted file would contain Figure 3 as a subdiagram.

In a sense, then, the new file uses the "inversion" idea in a more generalized form, but applies it in a more effective way in order to avoid repeated storage of the same information.

Detailed Intercomparison of System Efficiencies

Obviously a conclusive comparison of the relative efficiencies of the three file types is a matter for experimental test on large real files by using real requests. No such tests have been performed to date. However, several possible tests are under discussion at the National Bureau of Standards at the present time.

One interesting test could be made on chemical structures. Descriptors can be chosen either from the chemical elements or from the functional groups which can occur in the structures. Since the occurrence or absence of these in structures is unambiguous, the worst problems of finding a "fail-safe" vocabulary are avoided. The "library" indexed by these descriptors might have as "documents" single structures, sets of structures, and/or information about properties, depending upon the application envisioned.

In the absence of empirical knowledge we are reduced to the procedure of examining degenerate cases, which may be quite informative. The cases of interest are (a) a random sample of questions which elicit nearly the whole file as answers, and (b) a random sample of requests which reject nearly the entire file or the whole file. The details of the comparison depend, of course, upon the exact way in which each of the procedures is programmed. What one would like to know is how many instruction executions on some computer are required to process the average request. The best we can do here is to count "manipulations", but this provides a good indication.

In the case in which nearly the whole file answers the request, the two more conventional files are probably superior, but for different reasons. The descriptor inverted file will be the quickest for the simple reason that, given a sensible vocabulary, a request can have few if any descriptors in it if it is going to get anywhere near 100% response. Granted that only a few descriptors can occur in a high percentage request, at most two or so lists need be intercompared. For one, the job is done immediately; for two or three, by any efficient algorithm, probably fewer than M document identification-number comparisons are required.

In the case of the ordinary file, M descriptor or rejector lists must be compared with the request. If vectors are used, this amounts to M 'and' -and-test sequences (multiple precision if necessary).

In the case of the new file, we observe that every set of identification numbers (with the same rejector vectors) has a "last" $S_{jk\ldots p}$ associated with it alone. But there is then a superstructure of molecular rejectors with fewer subscripts erected over this base. There are no more than 2^N different rejector vectors possible, and the actual number in the file will often be somewhat smaller. If $M > 2^N$ by any great amount, it is possible that there are fewer than $M-2^N$ "super-structure" molecular rejectors. In any event, every request satisfied by nearly every document will have to be compared with nearly every molecular rejector's rejector list, and if $M \leq 2^N$ this will be more expensive than the ordinary file. (Note that if this is really done in list-processing form, rather than in "vector" form, time will be saved by the fact that there are fewer rejectors per molecular rejector than there are per individual rejector vector. In fact, as far as number of descriptor-rejector comparisons are concerned, the values will be comparable.)

Shifting to the opposite extreme now, we see the new type file come into its own. Here nearly every document is rejected. We observe that the single-subscript rejectors may be nearly all that need be examined. In the majority of requests having 100 percent rejection this will be true, assuming the sticky problem of finding a good measure of merit $\mu(S)$ has been solved satisfactorily. Unfortunately it is difficult to estimate the probable number of these single-subscript rejectors. It will obviously be between 0 and 2^N, and considerably less than the latter. If the degenerate "one atomic rejector per molecular rejector" case is considered, there would be N (or fewer); the number may lie near N, in any case. At least, it will be much less than M.

In some near-total rejection cases a request may filter fairly far down the "rejector tree" before being blocked, but because of the use of μ, this will be in the minority of cases. Hence the average number of molecular rejector M request comparisons will be small, generally much smaller than M, particularly if $M > 2^N$.

Here, as in the total acceptance case, the ordinary file still requires M request list-document list comparisons and will lose out so far as efficiency is concerned.

The inverted file is a bit more complicated. Suffice it to say that if a given document has k out of the r descriptors requested, it is rejected k times, in effect. Until a request descriptor is reached that does not apply to that document, the document appears in the "candidate list". Even after it is stricken from that list, other lists must be manipulated in which it is carried as excess baggage, and for each of these it must be classified as "not in the candidate list" and discarded. Since the molecular rejector file rejects each document only once (in a sense less than once since the document in question is likely to be rejected along with others), it is more efficient in this respect.

To summarize our results, then, the new file is likely to be superior to the ordinary file if the screen is powerful enough to average a very high percentage of rejection. If it is not this powerful, then it is hardly worth using in any case. If $M > 2^N$, it is better in all cases. We also conclude that the molecular rejector file is very probably better than the inverted file in the high-rejection cases, but there is a much stronger need for data in this case.

A Final Efficiency Problem

Even granting that a case can be made for the contention that the new type of file is superior to the other two if heavy screening is possible, a more difficult question remains. An estimate is needed of the savings which will result from use of the pre-processed file, as contrasted with the expense of writing, debugging, and then using the file-organizing routine. This question appears to be unanswerable without data, but it does suggest the desirability for a potential user to consider the "request-traffic" of his file. A file with light traffic will naturally take longer to pay for itself than one with heavy use. Because of the possibility of periodic reprocessing of the file because of new documents (as discussed before), the use must be heavy enough to pay for this, at least.

It is hoped that actual machine testing of some of the theories described here will be possible in the near future.

Conclusion

Two basic ideas helpful in constructing information retrieval systems have been discussed. First, the advantages of the screening approach to the file organization in the development of systems were argued. Secondly, the desirability of pre-processing files to take advantage of block-rejection was suggested. These two ideas were then applied to the problem of setting up a nearly fail-safe screen using descriptors or, more accurately, rejectors. An algorithm was sketched for generating descriptions of documents in terms of "molecular rejectors". Another algorithm was outlined for storing on tape a file of the type described. Also discussed was an alternative "storage" system, using the idea of "dynamic storage" in which information is conveyed by branch points in a program.

The new type of file was then compared with the better-known ordinary and inverted files. The conclusion of the theoretical reasoning was that the system shows enough promise of increased efficiency to warrant detailed testing of real files.

References

1. For the logical (propositional calculus) features: Harold Pfeffer, Herbert R. Koller and Ethel C. Marden, "A First Approach to Patent Searching Procedures on Standards Electronic Automatic Computer," American Documentation, Vol. X, No. 1, pp 20-26, Jan. 1959.

   For topological network features: Herbert R. Koller, Ethel Marden, and Harold Pfeffer, "The HAYSTAQ System: Past, Present and Future," Preprints of Papers for the International Conference on Scientific Information, Washington, D.C., Nov. 16-21, 1958. (Area 5, pp. 317-553.)

2. "Depth" refers to the degree to which the complete content of a document is represented by the descriptor set (or other representation) assigned to it. Deep indexing, then, leaves out very little of the content of any document in the file.

3. D. D. Andrews and Simon M. Newman, "Activities and Objectives of the Office of Research and Development in the U.S.Patent Office," Journal of the Patent Office Society, Vol. 40, No. 2, Feb. 1958, pp.79-85.

4. F.E. Firth, An Experiment in Literature Searching with the IBM 305 RAMAC, San Jose, California: IBM, November 17, 1958;

   J. J. Nolan, Principles of Information Storage and Retrieval Using a Large Scale Random Access Memory, San Jose, California: IBM, November 17, 1958.

5. Jacob Leibowitz, Julius Frome, and Don D. Andrews, Variable Scope Patent Searching by an Inverted File Technique, Patent Office Research and Development Reports...No. 14, U. S. Department of Commerce, Washington, D.C., Nov. 17, 1958.

   Jacob Leibowitz, Julius Frome, and F. D. Hamilton, "Chemical Language Coding for Machine Searching," Abstracts of Papers, 135th Meeting, American Chemical Society, Boston, 5-10 April 1959, p. 3G.

6. Victor H. Yngve, "The Feasibility of Machine Searching of English Texts," Preprints of Papers for the International Conference on Scientific Information, Washington, D.C., Nov. 16-21, 1958. (Area 5, pp. 161-169)

   ______, In Defense of English, Preprint of Paper presented at An International Conference for Standards on a Common Language for Machine Searching and Translation, Sept. 6-12, 1960, Cleveland, 8 p.

7. If tape is used, it is clearly most efficient to replace M_i with the number of words between M_i and the appropriate M_j, so that the tape can be advanced immediately without having to examine each word until M_j is found. For large files (many tapes),

tape number and tape position could be specified in place of M_1.

8. J. C. Shaw, A. Newell, H. A. Simon and T. O. Ellis, "A Command Structure for Complex Information Processing", Proceedings of the Western Joint Computer Conference - Contrasts in Computers, presented at Los Angeles, California, May 6-8, 1958, pp. 119-128.

John McCarthy, "Recursive Functions of Symbolic Expressions and Their Computation by Machine, Part I," Communications of the ACM, Vol. 3, No. 4, April 1960, pp. 184-195.

Victor H. Yngve, "The COMIT System for Mechanical Translation", Proceedings of the International Conference on Information Processing, UNESCO, Paris, France, June 15-20, 1959, pp. 183-187.

Appendix

Details of Selecting an Optimal File Structure

A. Introduction

Here we address ourselves to the details of the choice of an optimal molecular rejector language for the file. The desirability of this molecular description will be assumed.

The concept of an optimal language needs considerable amplification. We wish to minimize the average number of molecular rejector-request comparisons needed to "screen" a given request, assuming that the amount of machine time used in screening one request is proportional to the number of comparisons performed. This assumption will be strictly true for fixed field processing in which N is less than or equal to the number of bits in a word, and on the average true when fixed field "vectors" must be processed with multiple precision. It is not really valid if variable field (list-type) processing is used. In the latter case a different analysis is needed.

The notion of an average is necessary since, as the section on "Evaluation" points up, the number of comparisons for a given request is strongly dependent upon the nature of the request. But then an unpleasant question asserts itself: "Over what do you average?" The answer one would like to give is "We want the average over all the requests which will ever be presented to the file." If this average is minimized, we have the best file, by definition. Unfortunately this is an unhelpful answer to the question since it cannot be found until all requests have been processed. Then it is a bit late to be of any use.

Hence we must resort to imperfect predictors of what this average would be for each molecular vocabulary in order to choose the best one. Each prediction method can then be used in a vocabulary generation algorithm. The reciprocal of the predicted average may be used as the "quality" of the file.

However, another competing criterion is involved, that of the complexity of the algorithm needed to apply the prediction of averages to the choice of a vocabulary. Such an algorithm when programmed, must be able to find a "best" vocabulary in some reasonable running time. Unfortunately, in order to get manageable routines, we have to cut corners and make shaky approximations in various places. Ultimately, we must balance time saved in screening against time consumed in processing, and stop at some (at this time) ill-defined "break even" point.

B. Development of the Prediction Method

One can break down the prediction methods into two types: external sample methods and file sample methods. In the external sample methods, a sample of requests is gathered from potential users, and these are used as predictors of the kind of requests likely to be directed to the file. On the other hand, to the extent to which the content of the file is representative of the distribution of interest of people using the file, statistics about occurrence of molecular descriptors in the file can be used to predict requests without external data from potential users. Our tentative hypothesis is that this latter method is probably quite adequate. It is obviously cheaper, since it reduces the amount of error-free machinable data that must be prepared.

One quite serious problem in the choice of the best molecular vocabulary arises from the question of how many alternative complete vocabularies must be generated. If we have a method of assigning each vocabulary a value, clearly we could generate all possible vocabularies, assign each a value, and then choose the best. However, the number of operations involved grows as some "fierce" exponential of the number of documents in the file and the vocabulary size.

The most desirable method would follow a direct path from molecular rejector to molecular rejector, in every case choosing exactly that one which will lead to the best final file. What this requires is that at every point we be able to assign to each candidate-rejector S a "measure of merit" $\mu(S)$ which predicts the ultimate quality of the file which will result if it is chosen. It is by no means clear that such a μ can be found which avoids effectively "trying out" nearly every possible resultant file (another exponential factor). At this point, too, we will find it necessary to make imperfect predictions in order to find a manageable process.

Our procedure might be described as pessimistic, or cautious. When choosing a molecular rejector at any point, we consider the worst possibility for number of comparisons in a file determined by the choice, and then choose the one that gives the "strongest guarantee against the worst". In other words, at each point of choosing a molecular rejector, we are in a position to assert, "The maximum possible number of comparisons to which I might obligate myself if I choose this rejector is k. Then we choose the rejector with minimum k. Every rejector will, in fact, do better than this estimate when the whole vocabulary is chosen (this will be clarified later), and it is possible that one of the apparent underdogs would come out best in the end, but to check this, one would have to "try it and see", which means another case of exponential growth in number.

We reason as follows:

1. For each molecular rejector S there will be a probability P(S) that a "random" request will have one or more bits in common with S. This we will call the "rejection probability" of S, the probability that S will be able to reject all the documents to which it applies. How one computes P(S) will be taken up later.

2. For a (sub)file F with R different rejector vectors (documents with like vectors are already grouped), the number $t(S_i,F)$ of vectors to which a given S_i applies is calculated. The subfile to which it applies is F_i.

3. We then note that if a request Q is directed to the (sub)file F, it has a probability $P(S_i)$ of being rejected by S_i, and hence a probability $1-P(S_i)$ of passing on to the file F_i.

4. Being pessimistic, we assume that after S_i is removed, each of the F_i rejector vectors has to be given its own molecular rejector. Hence if Q "passes through" S_i, $t(S_i,F)$ further comparisons must be made. But only $(1-P(S_i))x10^2$ percent of the requests are passed through. This leads to an average of $(1-P(S_i))\cdot t(S_i,F)$ comparisons.

5. In addition, Q must be directed at the $R-t(S_i,F)$ other vectors. Assume all of these must be treated as rugged individualists, at one comparison per vector.

6. Totalling the number of comparisons (including that with S_i itself), we get $K=1+(1-P(S_i))t(S_i,F)+R-t(S_i,F)=R+1-t(S_i,F)\ P(S_i)$.

7. We observe that if $t(S_i,F)\cdot P(S_i)< 1$, then $K < R$. Since R comparisons would be involved if each vector were processed separately, in this case use of the molecular rejector S_i will be an improvement over "ordinary" processing. If $t(S_i,F)\cdot P(S_i) < 1$, all the vectors should probably be stored individually.

8. Furthermore, we note that if this reasoning is applied to the subfiles of F_i and $F-F_i$, respectively, it is clear that each of these will probably do better than the $t(S_i,F)$ and $R-t(S_i,F)$ ascribed to them, further justifying the fact that 4 and 5 are pessimistic estimates.

9. It is clear that we minimize the number of comparisons that the (sub)file F contributes to the total by maximizing $f_iP(S_i)$ where f_i is the number of documents in F_i. Therefore, we choose as the measure of merit $\mu(S_i)$ the number $\mu(S_i) = f_iP(S_i)$.

10. Suppose $\mu(S_i) = \mu(S_j)$. We would like to find a further criterion for choosing S_i or S_j as "most likely to succeed". Suffice it to say that the one of largest P(S) is probably best. If $P(S_i) = P(S_j)$, a random choice is the best we can do.

C. <u>Estimates of the Rejection Probability P(S)</u>

Clearly the tally $t(S_i,F)$ can be obtained simply by counting and presents no important problems. The rejection probability $P(S_i)$ is more difficult, however. A number of methods of evaluating or estimating $P(S_i)$ will be presented here, in order of decreasing sophistication (and complexity, in general).

1. <u>Outside Sample; Complete Request Method</u>. Here we assume that information is available from an outside sample. Every possible set of descriptors which might make up a request Q (all 2^N-1 of them) is tested against the sample, and a percentage P(Q) is computed. P(Q) is the percent of sample requests which are exactly the request Q, in other words, the probability that a random request will be Q. A table of the $P(Q_i)$ for all 2^N-1 of the Q_i is stored.

For each S_i, all those requests overlapping with S_i, identified as R_{ij}, are generated. Then we have $P(S_i) = \sum_j P(R_{ij})$. (This is the usual probability that at least one of a list of independent events will occur. In this case the "event" is the occurrence of <u>exactly</u> the particular R_{ij} as a request and hence is independent of that for any other R_{ik}, $k\neq j$, even if $R_{ij} \wedge R_{ik}$ has many 1-bits in it.)

2. <u>File Sample; Complete Request Method</u>. The procedure is exactly the same as above, except that the file itself is used as the sample. This is not a trivial process, since the file is encoded in terms of its rejectors rather than descriptors. It is best done for each Q_i by looking for <u>exactly</u> $\overline{Q}_i$ (complement of Q_i) in the file, counting the number of these to get $N(\overline{Q}_i)$, and then dividing by M to get $P(Q_i) = \frac{N(\overline{Q}_i)}{M}$.

If the file is uncomfortably large, one might select a manageable random subfile for this purpose.

3. <u>Atomic Request Method: Either Sample</u>. Here, for each atomic descriptor D_i, we simply evaluate the probability $P(D_i)$ that this

descriptor will be found in a given request. We then estimate the number $P(Q_j)$ of the above analysis by listing the set $\{D_k\}$ that occur in Q_j (symbolically, $Q_j = \underset{k}{V} D_k$). We say $P(Q_j) \approx P(D_1) \cdot P(D_2) \ldots P(D_k) = \Pi P(D_k)$ (iterated product). Then the formula for $P(S_i)$ is computed as in C.1. This computation is based upon the well known formula for the probability that a number of independent events will occur together. We know that, in fact, the occurrence of descriptors in a request will not be independent; they will tend to "cluster". C.1 and C.2 take this into account, but it is thrown away here. Clearly a considerable computation is avoided however.

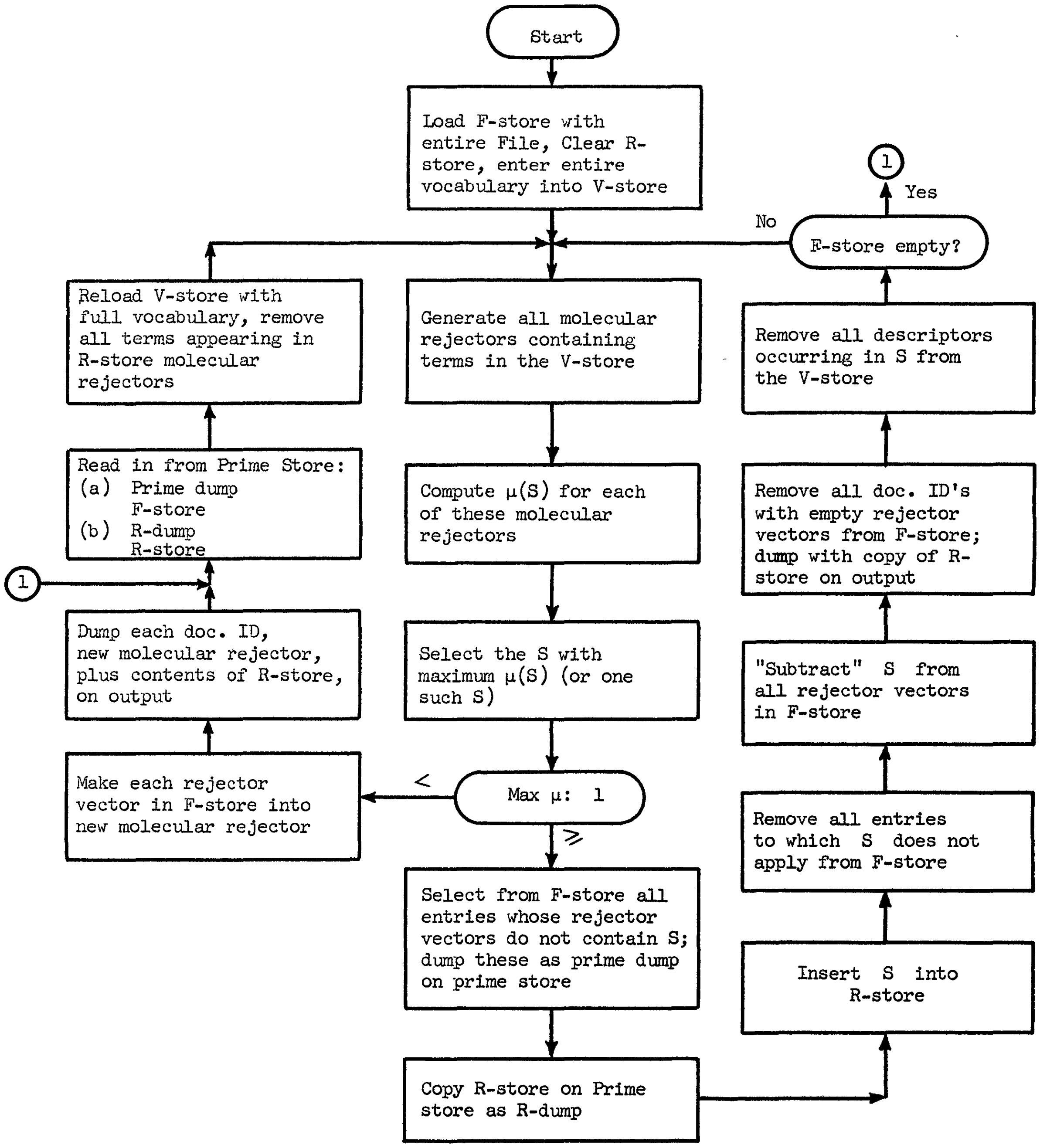
4. The No-Sample Complete-Request Method. Here we assume that all requests are equally probable. Thus we need only to evaluate the percentage of all possible requests which overlap S_i to get $P(S_i)$ to within some multiplicative constant (which can be ignored). There are $\sum N-1$ requests. If S_i has b_i bits, there are $N-b_i$ bits that do not "intersect" S_i and $2^{N-b_i}-1$ requests made up only of those bits. Hence there are $(2^N-1) - (2^{N-b_i}-1) = 2^N-2^{N-b_i}$ requests which do overlap with S_i. This $P(S_i)$ equals

$$C.\ \frac{2^N-2^{N-b_i}}{2^N-1} == C. \frac{1-2^{-b_i}}{1-2^{-N}} .$$

The important quantity is

$1- \frac{1}{2^{bi}}$, which clearly increases directly with b_i. (This can be found neatly on a computer with one-bit right shifts as one counts the number of bits in S_i and then subtracts.)

Clearly, all of these methods are programmable and not too forbidding when considered separately. The problem arises from the fact that the $\mu(S)$ subroutine will need to be used an astronomical number of times in generating a file structure and, in fact, is likely (it appears to the author) to account for a high percentage of the running time of the file-organization routine.


Start
Load F-store with entire File, Clear R-store, enter entire vocabulary into V-store
Generate all molecular rejectors containing terms in the V-store
Compute μ(S) for each of these molecular rejectors
Select the S with maximum μ(S) (or one such S)
Max μ: 1
<
≥
Make each rejector vector in F-store into new molecular rejector
Dump each doc. ID, new molecular rejector, plus contents of R-store, on output
1
Read in from Prime Store: (a) Prime dump F-store (b) R-dump R-store
Reload V-store with full vocabulary, remove all terms appearing in R-store molecular rejectors
Select from F-store all entries whose rejector vectors do not contain S; dump these as prime dump on prime store
Copy R-store on Prime store as R-dump
Insert S into R-store
Remove all entries to which S does not apply from F-store
"Subtract" S from all rejector vectors in F-store
Remove all doc. ID's with empty rejector vectors from F-store; dump with copy of R-store on output
Remove all descriptors occurring in S from the V-store
F-store empty?
No
Yes
1

Note: The Prime store is a storage area for primed subfiles. Each primed subfile is stored along with all those molecular rejectors applying to every document in the subfile. The left branch corresponds to Step 4 in text; the right to Step 5.

Figure 2. Flow Chart of File Organization Routine

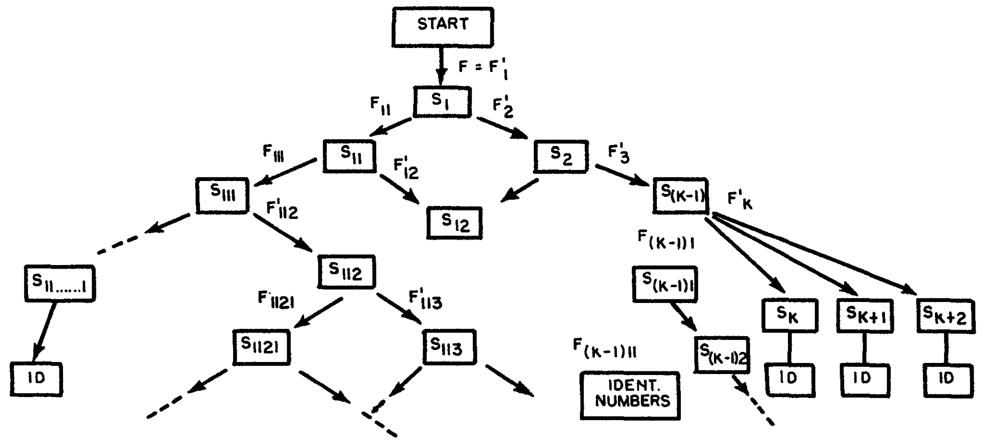
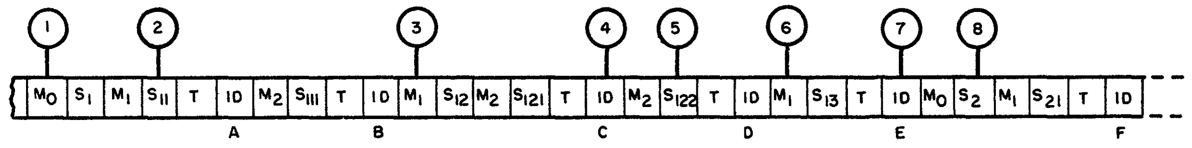
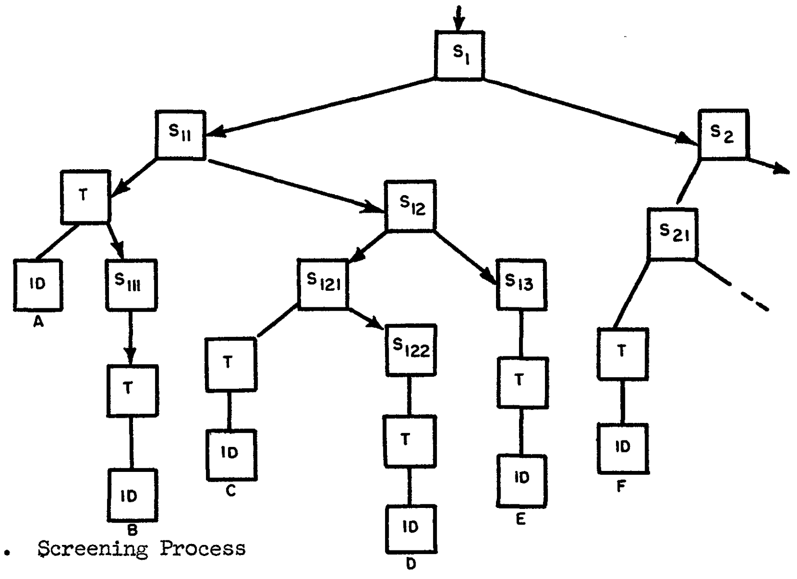


Figure 3. File Generation Tree

a. Tape Configuration



b. Generation Tree for File



c. Documents in File

Document	Molecular Description of Rejector Vector
A	$S_1 \vee S_{11}$
B	$S_1 \vee S_{11} \vee S_{111}$
C	$S_1 \vee S_{12} \vee S_{121}$
D	$S_1 \vee S_{12} \vee S_{122}$
E	$S_1 \vee S_{13}$
F	$S_2 \vee S_{21}$

d. Screening Process

Request Q has bits in common with S_{11}, S_{122}, and S_2, but with no other molecular rejectors, hence by c. should reject A, B, D, F. C and E should be accepted.

Processing starts at ①, proceeds word by word to ② where overlap is found, advances tape immediately to ③, proceeds word by word to ④, where print occurs, then on to ⑤, where overlap occurs. Tape is advanced to ⑥, where word-by-word procedure continues to ⑦, where print again occurs. At ⑧ overlap occurs, and tape is advanced to the next M_0 symbol, which will precede S_3.

Figure 4. Tape Configuration - Screening Process

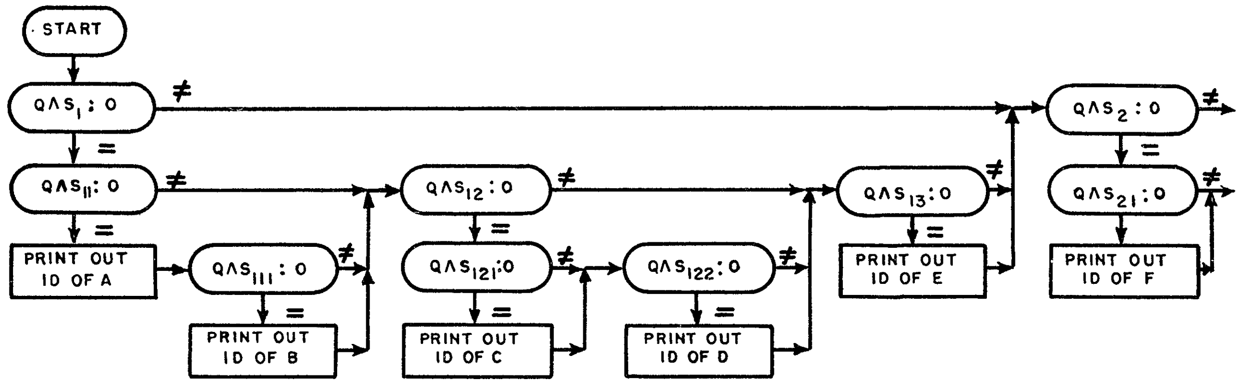


Figure 5. Flow Chart of Program-File
(Generation tree for this file is that given in Fig. 4b)

WHAT IS AN INTELLIGENT MACHINE?

W. Ross Ashby
University of Illinois
Urbana, Illinois

Summary

From the "intelligent" processes we must first split off those that are peculiar to the living brain, but only because they are not commonly met with elsewhere. These processes are of interest but are neither intelligent nor stupid, neither good nor bad.

The "intelligent" processes par excellence are the goal-seeking--those that show high power of appropriate selection. Man and computer show their powers alike, by appropriate selection. But both are bounded by the fact that appropriate selection (to a degree better than chance) can be achieved only as a consequence of information received and processed.

Machines can be made as intelligent as we please, but both they and Man are bounded by the fact that their intelligence cannot exceed their powers of receiving and processing information.

1. Introduction

I am very pleased to have the privilege of addressing this conference today because I believe the time has come when we should notice a turning point in our views on the nature of brain and of brain-like mechanisms. The 1950's were largely a decade of ferment and progress. The 1960's, I believe, will be a decade of consolidation, of the establishment of a firm framework of ideas within which the whole science of brain-like mechanisms will move for quite a number of years to come.

The point is that until recently we all tended to assume that the capacities of the brain, especially of the human brain, were unlimited. We felt that if a man were clever enough he could do anything--the genius could solve any problem. I say that that belief must go. It is obstructing progress. In the 60's it will be recognized as being as ignorant and superstitious as the belief of the small boy who thinks that his big brother can lift anything. Today, we know what the word "brain-like" means, and we know what are a brain's limitations. We know, too, that these limitations are exactly the same for the human brain and for the machine because they are the limitations inherent in any system that behaves in an orderly and law-abiding way. The system that passes these limitations gets its results by pure magic. Before I go on, however, to treat these matters in more detail, I would like to discuss some minor matters so that we can get them out of the way.

Brain-like processes can be clearly divided into two classes, according to whether the process is goal-seeking or not. It is the goal-seeking processes that are par excellence the intelligent ones, whether they occur in a machine or a brain. But there are also a number of processes that occur in the brain that are not goal-seeking. Let me deal with them first.

The living brain has, of course, a great number of interesting properties that are of interest simply because they do not occur commonly elsewhere. The brain, for instance, has some unique biochemical processes; and it has interesting electrochemical processes. Of special interest to the computing engineer are the special network properties that it has developed, and the stochastic properties that it has developed for special purposes. The chief point about these nongoal-seeking processes is that they are neither good nor bad in themselves--they are simply processes that the laws of nature provide--like oxidation--and the brain, under the guidance of natural selection and evolution, develops or suppresses them in accordance with whether they are useful or harmful. They are brain-like only in the sense that they are seldom seen outside the brain. They can all be simulated on computers because they are straightforward natural processes.

In considering these brain-like processes, we should remember that the computer is actually superior to the brain, because the computer can be made to behave as if it were totally devoid of any operational structure. As a result, the computer can, in principle, carry out any well defined process. The living brain, however, has been so molded by five billion years of evolution that it is now very highly specialized to match the needs of this terrestrial environment. This environment, we are beginning to realize, is nothing like so general as one is apt to think it. Its distribution in space, with a three-dimensional Euclidian metric, the extraordinary commonness of continuity in it, its tendency for effects to be much localized, and the tendency for the same properties to turn up again and again in different places, all these features are highly characteristic of the terrestrial environment. For them to occur in the computer, they would have to be programmed in with great labor. Because the living brain has faced this very special environment for so long, the brain has become equally specialized in its operational methods. As a result, the brain, far from being a remarkably flexible mechanism, is now appreciated as a system of remarkable inflexibility.

Of these brain-like processes, other than the goal-seeking, I wish to say little here, but before I leave the topic I would like to say that I think the most promising line of research at the present time is the study of systems with large numbers of equilibria. Our knowledge of such systems is extraordinarily small. A great deal is known about the statistical mechanics of large physical (i.e. Newtonian) systems, but these usually have very few states of equilibrium. The simplest example of a system with a really large number of states of equilibrium is a dish of sand, in which the particles will rest in a great number of different configurations. But the only activity of this system is the tiny movement of the grain of sand as it moves from a nonequilibrium to an equilibrium--a movement too small to be interesting. What we need to know more about is the system that has a very great number of states of equilibrium and, sufficient dynamism so that its trajectories, before it reaches a state of equilibrium, are sufficiently long and complex to be interesting. I have shown elsewhere[2] how such systems tend to show some of the elementary properties of living organisms, and I have little doubt that much more remains to be discovered. Here, of course, we recognize that a system with thresholds, such as is the nervous system is just such a system with a great number of states of equilibrium. It seems to me to be almost incredible that having known for fifty years that the nervous system works largely on threshold, we should, in the 1960's, know practically nothing about how such a system tends to behave when the system is large enough for it to show something of its statistical mechanics.

So much then, for those properties that the brain possesses, but that are essentially ordinary as natural processes. I now come to the other brain processes--those that are universally recognized as somewhat extraordinary. They are--the goal-seeking.

2. What Is "Intelligent"?

Until a few years ago there could have been a considerable dispute about what was meant by an "intelligent machine", but that time is past. The position was clarified some years ago, and has been known long enough for any refutation to have come forward. No refutation has been offered. Not a single clear counter-example has been given in the last ten years to show that an intelligent system is anything other than one which achieves appropriate selection. This is the touchstone of intelligence. According to this view, intelligent is as intelligent does.

Let me give some examples to make clearer what I mean. If a man plays chess, we need not judge his powers by listening to his boasting--we simply observe whether the moves he makes are very highly selected out of the totality of legal moves, being selected from just those few moves that bring him rapidly nearer the win.

Again, the good workshop manager is one who, in spite of all the confusions and difficulties of the day, issues such carefully selected instructions as will steer all the work through by the end of the day. Again, signal men show their intelligence by selecting just those patterns of operations in their box that gives, over long intervals of busy traffic, an accident number of zero. And in the so-called intelligence tests, which do test something of what we mean by intelligence, the operational criterion is simply "did the candidate select the right answers?"

Thus an intelligent machine can be defined as a system that utilizes information, and processes it with high efficiency, so as to achieve a high intensity of appropriate selection. If it is to show really high intelligence, it must process a really large quantity of information, and the efficiency should be really high.

In biological processes, appropriate selection and intelligence is shown essentially by regulation; the living organism, when it acts "intelligently," acts so as to keep itself alive. It acts, in other words, so as to keep the essential variables on which its existence depends within physiological limits. This is a straightforward act of appropriate selection, and the animals, as they ascend the scale of intelligence, show their ascent precisely by their power of regulating their environment in spite of greater ranges of stresses coming to them. In Man, the primary goals are what evolution and natural selection have built into him. The other goals are all secondary, developed either as species characteristics or by learning.

This approach to the nature of intelligence gives us an angle on the subject quite different from the older philosophers', and one which at a stroke ties it firmly to the modern theory of information. For "regulation" simply means that in spite of many threatened deviations from the optimum, the organism so behaves that the deviation does not occur; that is to say, the correct form is maintained. This achieving of a correct final form, repeatedly in spite of a stream of disturbances, is clearly homologous with the correction of noise by a correction channel. The noise threatens to drive the form or message from its desired shape and the correction channel so acts as to bring it back to the true form. A natural measure of the degree of intelligence can thus be given in the terms of Shannon's theory of communication, and with it not merely a measure but a complete grasp of the logic of the situation and of what is implied.

3. The Limit to Intelligence

As soon as we recognize that an intelligent system, whether living or mechanical, is simply one that behaves in an intelligent way, we appreciate that the test of intelligence is the power of appropriate selection. All intelligent actions are actions of appropriate selection. As a result every intelligent system is subject to the following postulate:

> Any system that achieves appropriate selection (to a degree better than chance) does so as a consequence of information received.

One would imagine this postulate to be completely obvious were it not for the fact that many discussions about the powers of living brains subtly and tacitly deny it. Yet what would happen if the postulate did not hold? We would have the case of the examination candidate who starts to give the appropriate answer before the particular question has been given! We would have the case of the man who submits an accurate insurance claim for damage by fire before the fire has broken out! We would have the case of the machine put on the market for which the claim is made that it is now so intelligent that it will start to give the answer before the program tape is run in!

Science knows nothing of these things. What will happen in the future we can't say; but it is quite clear that in the middle of the 20th century we must reject such possibilities and proceed on the assumption that they do not occur.

The moment we say such events do not occur we are implicitly saying that all systems whether human or mechanical are subject to this postulate--they can achieve appropriate selection only if they receive and process the appropriate amount of information.

This point of view at once brings them under a quantitative limitation. For appropriate selection is fundamentally homologous, as I said, with the correction of noise, and therefore the amount of correction that can be applied is subject to Shannon's tenth theorem[3]. Though the theorem has a somewhat different aim, it says that if a certain quantity of error is to be removed from the final form (that is to say, a certain degree of appropriate selection is to be made), then at least that quantity of information must be carried along the correction channel. When a human being undertakes such activities of correction, or of regulation, or of appropriate selection, he is acting as the correction channel, and he cannot achieve this appropriate selection unless he receives and transmits the necessary <u>quantity</u> of information.

The same point can be made in a simpler and more primitive form, as I have done in the law of requisite variety[1], which shows that in the most obvious and common-sense way the processing of the necessary quantity of information must be done if appropriate selection is to be achieved by law, and not by mere magic.

Today then, we are in the position of being able to say of the human brain that it must work in one of two ways. Either it works subject to this postulate, in which case it achieves appropriate selection because it has received and processed the necessary amount of information, or it is behaving in an entirely magical way, producing correct effects without corresponding causes.

I do not say that it is impossible that the human brain should sometimes do wonderful things--I believe that the universe is still full of surprises; but what I do say is that those who maintain that the human brain is not subject to my postulate must accept the consequences of the alternative and must declare that the human brain sometimes achieves appropriate selection without receiving the necessary information. And it is obviously desirable that they should produce evidence to show that this remarkable event does actually occur. Until such evidence is produced, the postulate must stand.

It may perhaps be of interest to turn aside for the moment to glance at the reasons that may have led us to misunderstand the nature of human intelligence and cleverness. The point seems to be, as we can now see with the clearer quantitative grasp that we have today, that we tended grossly to mis-estimate the quantities of information that were used by computers and by people. When we program a computer, we have to write down every detail of the supplied information, and we are acutely aware of the quantity of information that must be made available to it. As a result, we tend to think that the quantity of information is extremely large; in fact, on any comparable scale of measurement it is quite small. The human mathematician, however, who solves a problem in three-dimensional geometry for instance, may do it very quickly and easily, and he may think that the amount of information that he has used is quite small. In fact, it is very large; and the measure of its largeness is precisely the amount of programming that would have to go into the computer in order to enable the computer to carry through the same process and to arrive at the same answer. The point is, of course, that when it comes to things like three-dimensional geometry, the human being has within himself an enormous quantity of information obtained by a form of preprogramming. Before he picked up his pencil, he already had behind him many years of childhood, in which he moved his arms and legs in three-dimensional space until he had learned a great deal about the intricacies of its metric. Then he spent years at school, learning formal Euclidian methods. He has done carpentry, and has learned how to make simple boxes and three-dimensional furniture. And behind him is five billion years of evolutionary molding all occurring in three-dimensional space; because it induced the survival of those organisms with an organization suited to three-dimensional space rather than to any other of the metrics that the cerebral cortex could hold, evolution has provided him with a cerebral organization that must be peculiarly suited to the manipulation of three-dimensional entities. So when a mathematician solves a problem in three-dimensional geometry, he tends grossly to underestimate the amount of information involved in the process. When he does it in a computer, he tends grossly to overestimate it. What I am saying is that if the measure is applied to both on a similar basis it will be found that each, computer and living brain, can achieve appropriate selection precisely so far as it is allowed to by the quantity of information that it has received and processed.

Because of this hidden preprogramming of every human being, nothing is easier than for him to achieve results with extreme quickness, provided the question falls within his specialized range. But this is no more miraculous than the power of any other machine that is heavily preprogrammed to be quick. Most of the examples commonly given purporting to show some peculiar facility possessed by human beings are of problems in which human beings are peculiarly experienced, either personally or by the hereditary equipment that has come to them. Take for instance the playing of chess. The first thing that has to be explained to the boy of ten is that the rows, columns, and diagonals are significant. Because he is a human boy aged ten, long experienced in two-dimensional Euclidean geometry, we can indicate rows, columns, and diagonals to him by merely flicking a finger at the board. The computer, however, being stripped down to an absolute zero of metrical properties, has to have the whole metric of the chessboard explained to it in detail, because it would just as readily play on a board with a metric that would seem crazy and quite impossibly difficult to a human being. Thus the fact that a human being is especially good at human problems is no more remarkable than that the digital computer is especially good at problems involving powers of two, or that the analog computer is especially good at handling continuous functions. I say that whenever a human being is found to be peculiarly good at a particular class of problems he will always be found to have had substantial preprogramming in those problems. The alternative is that he is getting the answers by magic.

4. What Is a "Genius"?

We can now consider briefly the question of the so-called "genius," and the question of his nature. There are two gross fallacies that infest our thinking about the genius.

The first is that after many scientists have tried to solve a problem, we imagine that the one who solves it must have some peculiar power. This is about as reasonable as letting 1024 people predict how a coin will fall ten times in succession, and then, when one person gets all ten right, trying to find the explanation of his phenomenal powers of prediction!

Isaac Newton, for instance, recorded when he was quite young that he always thought of everything as flowing into everything else; this was just his natural way of thinking and very congenial to him. He used this way of thinking on almost everything. Is it surprising that this was the man, who, at a time when the calculus was on the verge of being discovered, was actually the man who got it first? Compare him with, say, Planck, at the beginning of this century, when science was crying out for a man who could think of everything as going in small, discrete jumps. Had Newton been unlucky enough to have been born in 1900 he would have found himself peculiarly handicapped at the time when the quantum theory was just being formulated. Clearly, the concept of a genius is apt to arise because after a number of workers have tried various ways of solving a problem, none of them knowing beforehand which is the right way, and one of them succeeds, <u>we</u> come along, pick this person out, and say he is remarkable. Now, part of the selection involved here was not made by that person; the selection is made by us, who pick out this person because of his performance. This very common mistake in statistical logic must be responsible for a substantial amount of our allocation of the title of "genius".

The second fallacy is the idea that the genius can go, as it were, straight to the answer without doing the work. In actual fact much of the work consists of making trials, which is, of course, a powerful way of gaining information. Many of the recognized geniuses are people who, by thinking about the subject day and night, are making trials of new combinations and new ways in great numbers. Take for instance the mathematician Gauss, who is doubtless generally accepted as an excellent example of a genius. Hear his own words about how he achieved a certain result, in a letter to Olbers: "Perhaps you remember my complaints about a theorem which had defied all my attempts. This lack has spoiled for me everything else that I found, and for four years a week has seldom passed when I would not have made one or another vain attempt to solve this problem — recently, very lively again. But all brooding, all searching, has been in vain." Then he adds, "Finally I succeeded a few days ago." And then he adds, "Nobody will have any idea of the long squeeze in which it placed me when I someday lecture on the topic." Undoubtedly, one of the reasons why a person is a genius is that he pays the price for it by sheer hard work. He processes the necessarily large quantity of information.

If the human brain is especially clever and slick at those problems for which it has been preprogrammed, we should find, of course, that it is peculiarly stupid and slow at those problems that are subtly contrary to the preprogramming. As far as I know, very little exploration has been done in this direction. We are not proud of our mistakes and it is only quite recently, almost within my lifetime, that psychologists have seriously paid attention to the defects of the ordinary human being instead of simply trying to exaggerate his abilities. But we do know that there are a certain number of events that show how he can be peculiarly handicapped. It is, of course, obvious that any species that tries to discuss its own sexual habits will always have difficulty, simply because the mixture of the real and the symbolic will always tend to create a confusion. We need not be surprised that we can discuss the sexual behavior of the stickleback with the utmost precision and objectivity, and then fall into a hopeless muddle when we try to talk about the sexual habits of the young man and woman of today. Other examples, are the well known difficulties that occur when we try to make simultaneous hand and foot movements in a way that do not match the age-old needs of gravitation and locomotion. Again, there are the demonstrations produced by Ames which show how strongly we are impelled to see relationships simply because we are preprogrammed to see them. There is one example, for instance, where one looks through a hole into a box and one sees apparently a toy chair suspended in mid air. Then one looks through a window in the side and one realizes that there are really a number of pieces scattered throughout the space but so arranged and strung on wires that when seen from one point they present the perspective of a chair. Then the observer, having seen beyond all question that the pieces are widely separated, goes back and looks through the hole again at the appearance. He <u>cannot</u> prevent himself from seeing one chair in one place.

5. There Is No "Real" Intelligence

Is there, then, no such thing as "real" intelligence? What I am saying is that if by "real" one means the intelligence that can perform great feats of appropriate selection without prior reception and processing of the equivalent quantity of information; then such "real" intelligence does not exist. It is a myth. It has come into existence in the same way that the idea of "real" magic comes to a child who sees conjuring tricks. At first the child believes in "real" magic. Later, after he has found how tricks are done he no longer believes in transcendental "real" magic; he replaces the myth by genuine knowledge of the processes of actual conjuring.

6. Consequences

What now are the consequences of this point of view? Especially for the computing engineer?

The first fact is that in talking about "intelligence," whether of the living brain or of the machine, we must give up talking about two sorts of intelligence. There is only one sort of intelligence. It is shown in essentially the same way whether the brain is living or mechanical. It shows itself by appropriate selection. It always implies the same underlying activity--that information in the required quantity is taken in (either immediately before the problem is given or at some time earlier as preprogramming) and that this quantity of information is processed with sufficient efficiency so that the total quantity does not fall below the point where it is no longer sufficient to allow the appropriate selection. The living brain has had only one problem throughout evolution: how to get the necessary information in, and how to process it with reasonable efficiency. The problem of today's computing engineer is exactly the same. From this point of view, the computing engineer should stop asking "How can I make an intelligent machine?" because he is, in fact, doing it at this very moment, and has been doing it for the last twenty years. He should stop being overawed by the so-called geniuses, and he should realize that the so-called genius is simply a rather extreme example of the system towards which he is working steadily. He will doubtless soon develop machines other than the large digital, but these will simply be intelligent in other ways. The contrast here is not between digital and analogue, or germanium vs protein, but between the true intelligence that processes information in due quantities, and the merely mythical intelligence that human beings have sometimes supposed themselves to possess.

A second application of the basic postulate is that it will provide guidance in a host of different processes. For one must realize that the rule about appropriate selection applies not merely to the final goal, but to all the subgoals that have to be found on the way to it, and to all the qualifying goals that may be set up. Thus, if the goal is a program to play good chess, the programmer may soon add a subsidiary goal:--the program is to be achieved in the shortest time. Now this "shortest time" is itself a goal, and its achievement demands appropriate selection (among the various ways that consume various times). Thus this demand itself can be met only by processing of the relevant information in sufficient quantity. If the information (about the relative quicknesses) does not exist in 1960, then the appropriate selection cannot be made. If the goal (of the quickest way) is still desirable, there is no way but that information must be collected. This means that there is no other way than that the program writer should try a tape, see how long it takes, try another tape, see how long it takes, and either by trial and error (another name for "experiment") or in any other way available to him, get the information about which way is the quickest.

There are a host of these subsidiary questions usually coming up during any real appropriate selection, and they can be extremely troublesome. It is a part of what I am saying that the basic postulate will apply to all of these subsidiary questions.

Any attempt to achieve a major goal usually implies the achieving of many minor or qualifying goals. The basic postulate, and the law of requisite variety cover them all. For example, to conduct a search quickly (with speed as the qualifying goal) may require repeated dichotomization. Then "how to dichotomize" becomes the object of a search. Finding how is an act of appropriate selection; it is again subject to the postulate.

There is again the problem of where to bring back corrective feedbacks: Should the correction be fed back to this point or to that? To know which is to make an act of appropriate selection, and this again can be done only insofar as information exists. If it does not exist either a simple random decision must be made or further information must be obtained, either systematically or unsystematically by trial and error; and so on, again and again.

To sum up. What is often referred to with bated breath as "real" intelligence, is a myth. The human being saves himself from being wholly foolish by having a great deal of information, as preprogramming, derived from millions of years of evolution on this earth, and by his personal experience over decades. Give him a problem in this range and he is really slick. This is his real intelligence. And any machine equally preprogrammed has an equal amount of real intelligence.

But this intelligence, whether of man or machine, is absolutely bounded. And what we can build into our machines is similarly bounded. The amount of intelligence we can get into a machine is absolutely bounded by the quantity of information that is put into it. We can get out of a machine as much intelligence as we like, if and only if we insure that at least the corresponding quantity of information gets into it.

The thought of this ultimate limitation is sobering, but the situation here today is not unlike that of power engineering a century ago.

At that time, so many powerful machines were being developed that many engineers took it for granted that the perpetual motion machine would soon be discovered. Then gradually emerged the idea that energy could not be created; and I have little doubt that this idea was seriously disappointing to many of the engineers of the time. They regarded it simply as a limitation.

Nevertheless, we now know that those engineers who accepted the limitation were in fact more realistic than those who went on hoping that perpetual motion would be possible. In the long run, the engineers who accepted it built better engines than those who went on struggling after perpetual motion. I suggest that today the position in computing is similar. If we accept the limitation--that appropriate selection can be achieved only to the degree that information is received and processed--and if we accept that this limitation holds absolutely over all brains, human and mechanical, our work, though less intoxicating, will in fact be more realistic. Those who build intelligent machines on this basis will outdistance those who want to build them on the old and superstitious basis that the human brain can do anything.

References

1. Ashby, W. Ross. An Introduction to Cybernetics. John Wiley and Sons, New York, 1956.

2. Idem. The Mechanism of Habituation. In N.P.L. Symposium on The Mechanization of Thought Processes. Her Majesty's Stationery Office, London, 1959.

3. Shannon, C. E. and Weaver, W. The Mathematical Theory of Communication. University of Illinois Press, Urbana, 1949.

ANALYSIS OF PERCEPTRONS

H. D. Block

Cornell University, Ithaca, New York

Summary

Perceptrons are self-organizing or adaptive systems proposed by Frank Rosenblatt as greatly simplified models for biological brains. The main objective is to begin to explain how the brain performs its functions in terms of its structural components. Consequently the methodology consists largely of investigating the behaviour of neural networks which, except for oversimplification, are not unreasonable models of the brain structure, and searching for non-trivial psychological behaviour. This is in contrast with the customary engineering approach of first deciding what function is to be performed and then designing a system to perform the desired function.

A perceptron is a network consisting of ideal neurons, similar to those of McCulloch and Pitts, connected together more or less at random, subject to certain organizational constraints and laws of growth. In this paper we give a brief introduction to the subject and describe some of the results of the mathematical analysis of several such systems; namely 'simple' perceptrons and 'four layer series-coupled' perceptrons.

Introduction

In 1956 Frank Rosenblatt proposed an adaptive or self-organizing system which he called a perceptron.[1] The principal purpose was to provide a model which would begin to explain how the brain performs its functions in terms of its structure. It was intended that the system would be essentially consistent with the known facts of neurophysiology, except of course in the direction of oversimplification. The model was defined with sufficient precision to offer the possibility of exact analysis and prediction of its behaviour. It was hoped that this behaviour would exhibit some aspects of perception. This hope was soon justified, and the perceptron attracted interest as a pattern recognizing device. Although some such applications are being studied,[2,3] the main orientation of the current research program at Cornell University is toward the brain function-structure problem. Consequently, the methodology of this research differs from that of an engineering program directed toward the development of a pattern-recognizing device, in the following way. The engineer designing a pattern-recognizer would start with the class of patterns to be recognized. He would then construct a system which would, when presented with each of the patterns, produce the desired response. In brief, the engineer starts with the function he wishes the machine to perform and then creates the structure that will perform that function. Rosenblatt's approach is somewhat the opposite. He starts with a system consisting of neurons, similar to those of McCulloch and Pitts, connected together more or less at random subject to certain organizational constraints and laws of growth. After the system has been specified it is then studied to determine how it will function when presented with stimuli. Consequently there exists a difference in viewpoint and methodology. We emphasize this point here, since it explains the formulation and approach employed in the problems to be described below. [Admittedly, a system which exhibits no interesting behaviour will be discarded, so that a certain amount of "design" is involved in the selection of the systems to be investigated. Similarly, the engineer will systematically check his synthesis by analysis. Thus the contrast should not be pushed too far; it is really a question of emphasis.] A somewhat more detailed exposition of the background of perceptron theory can be found in Reference 4, and a rather thorough discussion in Rosenblatt's comprehensive survey report.[5]

Perceptrons

A perceptron is a system of the general type indicated in Figure 1. A stimulus activates a subset of the sensory elements. The activated sensory elements send impulses, with various time delays, to the associators. Some of the impulses are positive (excitatory) and some are negative (inhibitory). If the algebraic sum of the impulses arriving at an associator in a suitable time interval exceeds a certain threshold (which need not be the same for all associators) that associator becomes activated and sends out impulses, as indicated by the arrows, to other associators and/or to the response units. The amount of impulse carried by a connection (the 'value' of a connection) varies in accordance with a 'reinforcement rule'; for example a connection whose tail and head are sequentially active is reinforced so that the impulse transmitted by this connection will tend to be increased. This mechanism furnishes the 'memory' of a perceptron. After being activated a unit might suffer a "refractory period," during which it cannot again be activated. Similarly, the response units have an activation threshold and connections with associators and/or each other.

Parameters which must be specified to define the perceptron of Figure 1 are: The number of sensory elements, the number (or probability distribution) of excitatory and inhibitory connections and the geometrical constraints on them, the number of associators and the number of response units; also the thresholds, refractory periods (if any), summation intervals and transmission times. The reinforcement rule and the initial values of the connections must be specified.

For studying the behaviour of such a perceptron it is necessary also to specify the set of stimulus patterns; the order and times of their presentation and the observations to be made on the responses.

The specification of the above items may be given in terms of probability distributions.

These systems are defined in more precise terms in Reference 5. In the interest of brevity we shall not discuss here the consistency of the above model with the biological constraints. The reader interested in this question will find a thorough discussion in Reference 5.

The behaviour of such systems has been the object of considerable study by mathematical analysis, simulation on digital computers and by experiments with a simple hardware perceptron. The results to date are presented in Reference 5. A system of the generality described above presents formidable problems of analysis. For only a few special types is the analysis more or less complete. We describe some typical results in the next two sections.

Simple Perceptron

A simple perceptron is indicated in Figure 2. Let us denote typical sensory units by s_σ, typical associators by a_μ, and typical stimuli by S_i. Let us represent the connection between s_σ and a_μ by the real number $C_{\sigma\mu}$; in particular the $C_{\sigma\mu}$ might be random numbers having the possible values + 1, - 1, 0. When the stimulus S_i is applied to the retina, the signal

$$\alpha_\mu^{(i)} = \sum_{s_\sigma \epsilon S_i}^{\sigma} C_{\sigma\mu}$$

is transmitted instantly to the associator a_μ. If

$$\alpha_\mu^{(i)} \geqq \theta$$, where θ is an arbitrary

but fixed real number, the associator a_μ is said to be <u>active</u> and instantly transmits a signal v_μ to the response unit. An inactive associator transmits no signal. The total signal arriving at the response unit is

$$u^{(i)} = \sum_\mu^{(i)} v_\mu$$, where $\sum_\mu^{(i)}$ is taken

over the associator units activated by S_i. If $u^{(i)} > \Theta$, where Θ is an arbitrary but fixed non-negative number, the response output is + 1. If $u^{(i)} < -\Theta$ the response output is - 1. If $|u^{(i)}| \leq \Theta$ the response output is 0.

Let us assign each stimulus S_i $(i=1,2,\ldots,n)$ to one of two classes which we denote by + 1 and - 1. Say stimulus S_i is assigned to class ρ_i where ρ_i is +1 or - 1. This dichotomization is then represented by $\rho=(\rho_1, \rho_2,\ldots, \rho_n)$. The response of the perceptron to stimulus S_i is said to be correct if, when S_i is presented, the sign of the output is ρ_i. We say that a solution exists to this discrimination problem for this perceptron if there exist numbers y_μ such that if $v_\mu=y_\mu$ then the perceptron will give the correct response to all the stimuli.

The 'error correction' reinforcement procedure is as follows. A stimulus S_i is shown, and the perceptron gives a response. If this response is correct then no reinforcement is made. If the response is incorrect then the v_μ for active associators a_μ is incremented by $\eta\rho_i$. The inactive associators are left alone. The initial values v_μ are arbitrary. The stimuli are presented in any order S_{i_1}, S_{i_2},... with repetitions allowable.

Theorem. Given a perceptron of the type described above and a dichotomy for which a solution exists, then there is a constant M such that, under the 'error correction' reinforcement procedure described above, the response of the perceptron will be incorrect at most M times. In particular if each stimulus recurs infinitely often then the perceptron, after a finite number of stimulus presentations, will thereafter identify all stimuli correctly.

In words: If a solution exists the perceptron will learn the dichotomy in a finite number of steps.

For proof, generalizations and analysis of other reinforcement rules see References 4 and 5. Although a few questions regarding the simple perceptron remain unanswered, the theory has reached the point where performance of many such systems can be closely predicted and, conversely, values of the parameters can be specified which will insure successful performance of prescribed discrimination tasks. These systems have a limited ability to generalize. Only in exceptional cases do they exhibit meaningful spontaneous classification of patterns.[5]

Four Layer Series-Coupled Perceptrons

The simple perceptron of Figure 2 generalizes on the basis of overlap of the stimulus patterns on the sensory field. The perceptron of Figure 3 generalizes rather on the basis of temporal contiguity of the patterns.

The values of the S to A^I connections do not change with time. The A^{II} units are in one-to-one correspondence with the A^I units and have threshold Θ. An active A^I unit a^I_μ delivers a fixed signal of Θ to its corresponding A^{II} unit a^{II}_μ and also a time dependent signal $v_{\mu\nu}$ to a^{II}_ν $(\nu=1,2,\ldots,N_a)$, where N_a is the number of A units. An inactive unit puts out no signal. The values $v_{\mu\nu}$ are initially zero and change with time as follows. Stimuli are presented at times 0, Δt, $2\Delta t$, $3\Delta t$,... . If a^I_μ is active at time t and a^{II}_ν is active at time $t + \Delta t$ then $v_{\mu\nu}$ receives an increment $(\eta.\Delta t)$; otherwise it does not receive this increment. At the same time each $v_{\mu'\nu'}$ is decremented by $(\delta.\Delta t)\ v_{\mu'\nu'}$. These two effects represent a facilitation of used pathways and a decay, respectively. The A^{II} to R connections have values which may be varied according to one of the standard rules of reinforcement as in the simple perceptron of Figure 2. For convenience we take these initial values to be zero, so there is no activity in the R units until the experimenter intervenes. Since we are interested in the self-organization of this system in the presence of an organized sequence of stimuli, this intervention of the experimenter will not occur, as will be seen below, until the experiment is almost over. There is no time delay of transmission of signals through the system.

The analysis of this system is

given in detail in References 5 and 6. Here we shall describe some of the results of that analysis.

Let $\gamma_\nu^{(i)}(t) = \sum_\mu^{(i)} v_{\mu\nu}(t)$, where $\sum_\mu^{(i)}$ is taken over those μ such that the associator a_μ^I is activated by S_i. That is, $\gamma_\nu^{(i)}(t)$ is the input signal from A^I units a_μ^I ($\mu \neq \nu$) arriving at the A^{II} unit a_ν^{II}, at time t. Let $\beta_\nu^{(i)}$ be the input signal to a_ν^{II} from a_ν^I when stimulus S_i is presented. Let the probability of occurence of stimulus S_i be P_i and the probability of transition from S_i to S_j be P_{ij}. We assume these to be stationary. We also assume $\Delta t \ll 1$. Now it is shown in Reference 6 that for t sufficiently large the system response reaches a steady state. In this state the $\gamma_\nu^{(i)}$ are the unique minimal solutions of the equations:

$$\gamma_\nu^{(i)} = \frac{\eta}{\delta} \sum_j \sum_k n_{ij}^I \, P_j \, P_{jk} \, \phi(\beta_\nu^{(k)} + \gamma_\nu^{(k)})$$

where n_{ij}^I is the number of A^I associators activated by both S_i and S_j; and $\phi(x)=1$ for $x \geqq \theta$, $\phi(x)=0$ for $x<\theta$.

From this basic result conclusions can be drawn about the terminal state of many such systems. We illustrate with two training programs on such perceptrons.

In the first training program the stimuli $S_1, S_2, \ldots, S_n$ are divided into two classes: $[S_1, S_2, \ldots, S_K]$ is class X, while $[S_{K+1}, S_{K+2}, \ldots, S_n]$ is class Y. There is assumed to be no appreciable difference in the retinal overlaps: $n_{ij}^I = q + s\,\delta_{ij}$, where $s>0$, $q \geqq 0$. Thus the diagonal elements of the n_{ij}^I matrix are all $(q+s)$ and all other elements are q. Note that by raising thresholds of the A^I units, the ratio q/s can be made as small as desired. The stimuli are presented at random, subject to the constraint that the probability of transition to a stimulus of the same class is p, nearly one, while the probability of transition to a stimulus of the opposite class is (1 - p), nearly zero. Inside a class all members are equally likely.

Let $A_o^{II}(S_i)$ be the subset of A^{II} units activated by stimulus S_i in the initial state and $A_{\infty}^{II}(S_i)$ the subset activated in the terminal state. If the parameters satisfy the inequalities

$$\frac{K^2}{sp+qK} \leqq \frac{\eta}{2\theta\delta} < \frac{K}{s(1-p)+qK}$$

then if $1 \leqq i \leqq K$

$$A_{\infty}^{II}(S_i) = \bigcup_{1 \leqq j \leqq K} A_o^{II}(S_j)$$

while if $K < i \leqq n$ then

$$A_{\infty}^{II}(S_i) = \bigcup_{K < j \leqq n} A_o^{II}(S_j) \quad .$$

That is, in the terminal state all stimuli of class X activate precisely the same set of A^{II} units, namely the set consisting of all those units initially activated by <u>any</u> stimulus of class X. Similarly each stimulus of class Y activates all those units initially activated by any stimulus of class Y and only these.

Thus the machine has dichotomized the classes, its codification being in terms of this intrinsic code on the A^{II} units. The experimenter has had no part in this experiment up to this point, except for his presenting the stimuli with the specified transition probabilities. If he now intervenes to give a single corrective reinforcement to the R units for one stimulus of each class the perceptron will then yield the correct response for all the stimuli. A similar analysis holds for more than two classes of stimuli.

The above result can be restated in the following, possibly more descriptive, terms. Here we take non-zero values on the A^{II} to R connections. The perceptron is shown a random sequence of letters of the alphabet, each letter occurring in various forms, fonts, and positions. The sequence is composed in such a way that a given letter, "A", is more likely to be followed by another form or position of the same letter, "A", than by a different letter. Ultimately, the perceptron will have seen a number of "runs" of each letter of the alphabet, each such run consisting of a sample of possible positions and variations. At the end the machine should assign a distinctive response to any letter presented; one response for "A"'s and another for "B"'s, etc. Of course, the particular assignment of responses cannot be specified in advance, since at no time does the experimenter give the machine any instructions based on his knowledge of what the letters are; he merely shows it one letter at a time, distorting and transforming it. It is not the topological similarity of the "A"'s with each other, nor the point-set overlap that is crucial here, but rather the fact that the "A"'s occur contiguously in time. Thus any set of objects that occur contiguously in time can be classified separately from any other sets whose members have the same property.

For the second training program consider the stimuli $S_1, S_2, \ldots, S_K$ and their transforms $S_{K+1}=T(S_1)$, $S_{K+2}=T(S_2)$, $\ldots$, $S_{2K}=T(S_K)$ under some one-to-one transformation T of the retinal points. For example $S_1, \ldots, S_K$ may be in the left half of the field and T a transformation which moves them to the right half. S_x, $(x=2K+1)$ is not shown during the training but is a test stimulus to be applied after the perceptron is trained. $S_y=T(S_x)$, $(y=2K+2=n)$. Let us assume S_x intersects $S_1, \ldots, S_L$, $(L<K)$, to a larger extent than it does the others. Specifically (cf. Figure 4)

$$n^I_{xj}=\begin{cases}(q+s\,\delta_{xj}) & j > L\\ (q+r) & j \leqq L\end{cases}$$

$$n^I_{yj}=\begin{cases}q & j \leqq K\\ (q+r) & K+1 \leqq j \leqq K+L\\ (q+s\,\delta_{yj}) & j > K+L\end{cases}$$

We also assume that no associator is activated by more than μ of the stimuli $S_1, S_2, \ldots, S_K$, where $\mu < K/L$.

A stimulus S_i from $\{S_1, \ldots, S_K\}$ is picked at random and the next stimulus is the transform $T(S_i)$. Then another is picked at random from $\{S_1, \ldots, S_K\}$ and this is followed by its transform, and so on.

If the parameters satisfy the inequalities

$$q(K+\mu)+\frac{Lr\mu}{K} < \frac{2K\delta\theta}{\gamma} \leqq q+r$$

then it is shown in Reference 6 that

$$A^{II}_{\infty}(S_x)=A^{II}_{o}(S_x)+\bigcup_{j\leqq L} A^{II}_{o}(T(S_j)\ ),$$

and

$$A^{II}_{\infty}(S_y)=A^{II}_{o}(S_y).$$

From this it follows first that when the machine has reached its terminal state the stimulus sequence S_x followed by S_y is characterized by a <u>decreasing</u> amount of activity, while the sequence S_y followed by S_x would yield an <u>increasing</u> pattern of activity. By connections having a time delay to the response units the machine can thus distinguish between a motion to the left (decreasing activity) and a motion to the right (increasing activity).

A more important result is this: The test stimulus S_x generalizes to its transform, even though neither one has occurred during the training sequence. This is the effect which was originally predicted for cross-coupled perceptrons,[7] and has since been demonstrated in digital simulation experiments.

Thus in the terminal state, after training with the transformation applied to unrelated stimuli, the machine, when taught the response to S_x and to another stimulus S_z, automatically gives the same response to $T(S_x)$ as it does to S_x and the same response to $T(S_z)$ as to S_z.

This result can be worded in perhaps more descriptive language as follows. The first stimulus is S_1, a random blob located in the left half of the sensory field. The second stimulus is $T(S_1)$, the same blob moved rigidly to the right half of the field by a translation, T, of a fixed number of retinal spaces. The third stimulus is another random blob S_2 in the left half of the field; the fourth is $T(S_2)$, its transform into the right half. The sequence is continued with random blobs in the left half of the field immediately followed by their transform to the right half, under the fixed transformation T. Then the system ultimately reaches a steady state in which, if the parameters are suitably chosen, the following behaviour will be exhibited:

In the terminal state, the machine is shown an A on the left and taught (for example by the error correction reinforcement procedure, described in connection with Figure 2, applied to the A^{II} to R connections) to give the response R_A. Then it is shown a B on the left and taught to give the response R_B. Now, when it is shown an A on the right (a stimulus it has never seen before), it gives the response R_A. When shown a B on the right (a stimulus it has not seen before), it gives the response R_B. Thus it has learned to 'identify as equivalent' (give the same response to) two patterns which are equivalent under the transformation T (and similarly T^{-1}); it has learned the transformation. The selection of T as a horizontal translation was for purposes of illustration; the result remains true for any one-to-one transformation. The use of blobs in the training sequence, rather than completely random pepper and salt patterns, is essential however, since with the pepper and salt patterns the same effect would be much more difficult to produce. This was already observed earlier.[7]

Other training procedures, in particular the symmetrical one in which either the stimulus or its transform can be the initial stimulus of the pair, are discussed in Reference 5.

Cross-Coupled Systems

The analysis of the general perceptron of Figure 1 presents several additional complications.

a) Closed loop reverberations are possible. Activity can go on independently of the stimuli presented and even if all stimuli are removed. The question of whether these reverberations die out, stabilize, or spread to activate all units, is crucial.

b) The set of A^I units activated by a given stimulus is no longer constant in time. This complicates the analysis. Moreover, they depend on the sequence of stimuli preceding the current one, rather than on the current stimulus alone. However, with suitable modifications, an analysis analogous to that used on the four layer system has been carried through.[5] The application of these results to specific situations is being studied further.

One further result of the above analysis which might be of interest to the engineer, is that the performance of these systems is relatively insensitive to malfunction or extirpation of a considerable fraction of the components. This means that as the system gets larger, the common (and often impossible) requirement for increased reliability of all components, is not encountered here.

References

1. Rosenblatt, F. The Perceptron: A Theory of Statistical Separability in Cognitive Systems. Cornell Aeronautical Laboratory, Report No. VG-1196-G-1, January, 1958.

2. Murray, A. E. Perceptron Applicability to Photointerpretation. Cornell Aeronautical Laboratory, Report No. VE-1446-G-1, November, 1960.

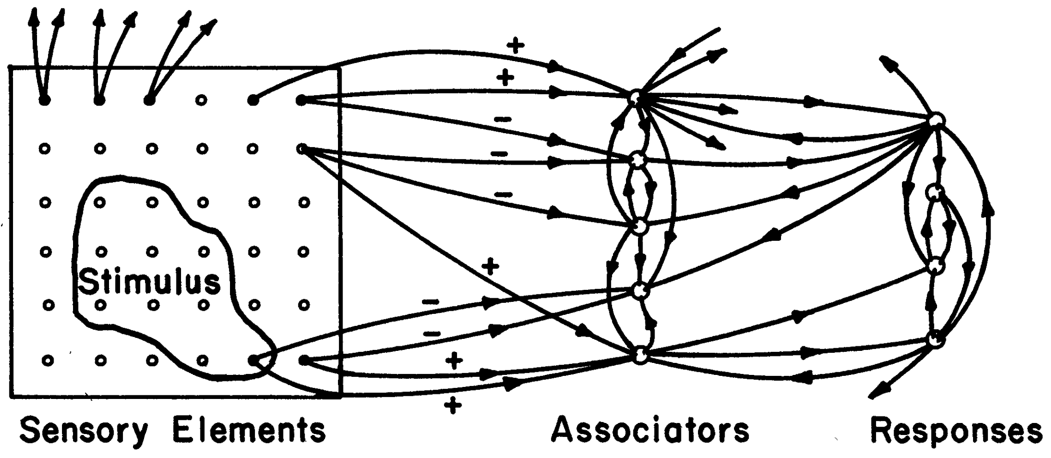
3. Kesler, Carl Preliminary Experiments on Perceptron Application to Bubble Chamber Event Recognition. Cognitive Systems Research Program, Cornell University, Ithaca, N. Y. Report No. 1.

4. Block, H. D. The Perceptron: A Model for Brain Functioning. Cognitive Systems Research Program, Cornell University, Ithaca, N. Y. Report No. 1.

5. Rosenblatt, F. Perceptrons and the Theory of Brain Mechanisms: Cornell Aeronautical Laboratory, Report No. VG-1196-G-8. September, 1960.

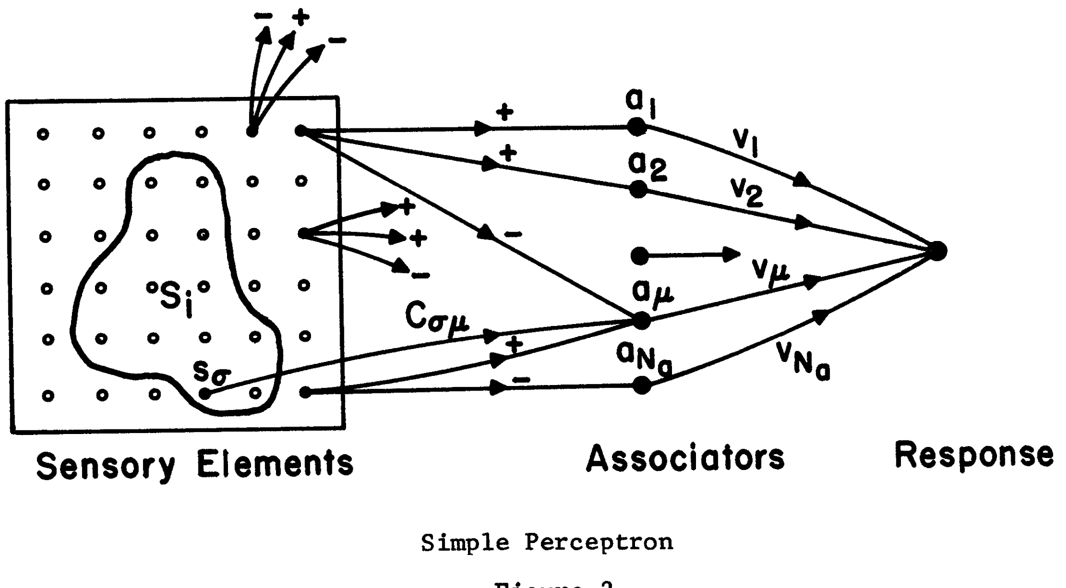
6. Block, H. D., Knight, B. W., Jr., and Rosenblatt, F. Analysis of a Four Layer, Series-Coupled Perceptron. Cognitive Systems Research Program, Cornell University, Ithaca, N. Y. Report No. 1.

7. Rosenblatt, F. Stimulus Generalization Over Transformation Groups. In Yovits and Cameron (Ed.), Self-Organizing Systems, Pergamon Press, London, 1960.



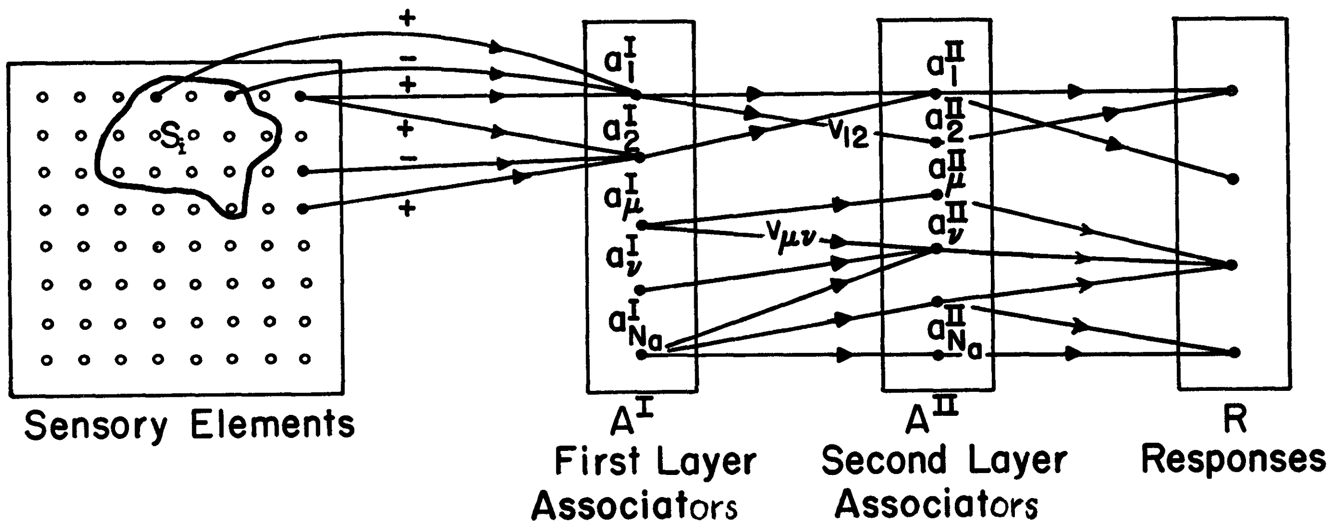
Organization of a Perceptron

Figure 1



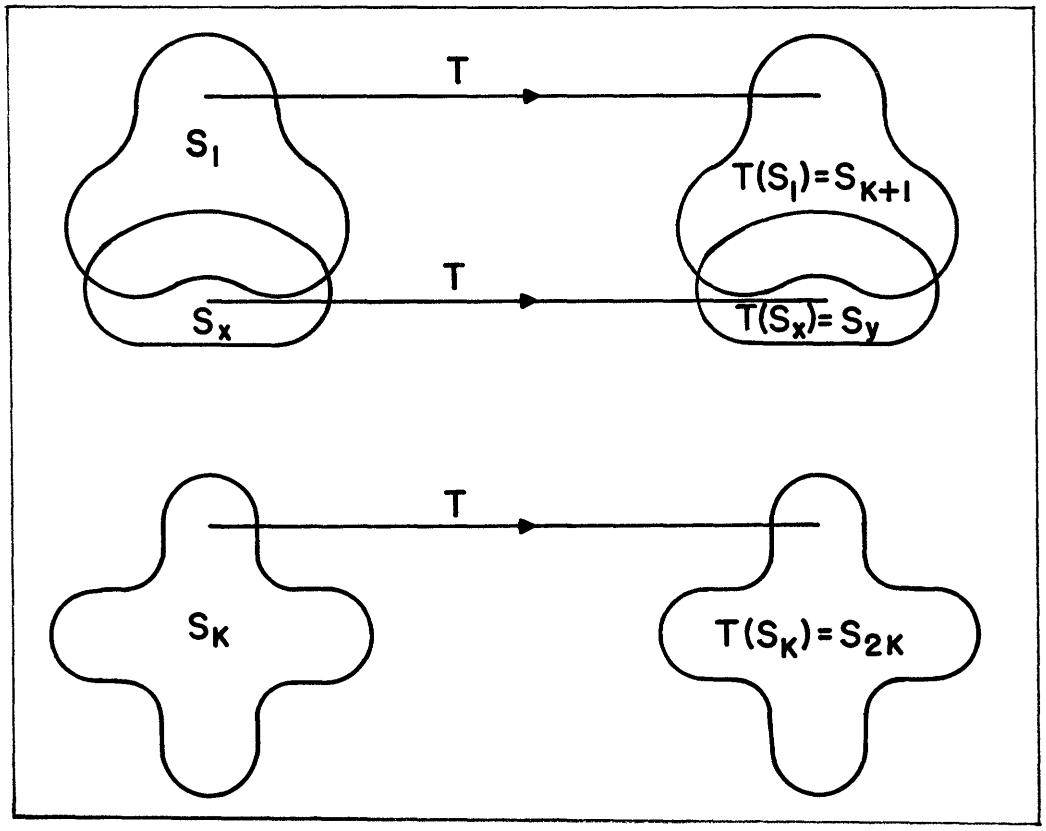
Simple Perceptron

Figure 2



Organization of Four Layer Series-Coupled Perceptron

Figure 3



Stimulus Patterns on Sensory Field

Figure 4

SOME PHYSIOLOGY OF AUTOMATA

Murray L. Babcock
Biological Computer Laboratory
Electrical Engineering Department
University of Illinois
Urbana, Illinois

Summary

Automata may make use of several methods of information and data reduction before classification or perception of the input stimulus pattern. Some methods may involve retaining as much information as possible throughout the process until the final classification is accomplished. Suggested here is a concept of pre-organization of information by "property filtration" which reduces the information as near the source within the automata as is feasible with respect to the over-all system compatibility. "Property filtration" is explained and illustrated by describing two practical and several theoretical property filters. The results of this work to date indicate that this approach to data reduction and pattern identification may greatly simplify the ultimate construction of adaptive automata, if such property filters are used as functional input units.

1. Introduction

Any automaton worthy of being considered a member of the general species will surely consist of an aggregate of functional units, each unit functionally supporting and perhaps augmenting the operation of all of the other units and of the automaton in general. To direct the total activity of all the units and in general to control the operation of the automaton, either an external control entity or some internal control entity will be used. This control unit may be either a single unit or may be a combination of units--in any case, however, this may be called a control center. This control center will place some value judgment upon each unit's operating ability and necessity, the judgment being solely determined upon the merits as to whether the well-being of the automaton as a whole is maintained. To enable the control center to make these judgments and control decisions, information from the various units of the automaton and from its environment must be presented to the control center in some acceptable form. In all but the trivial instances, the information as utilized by the control center will not be of the same form or of the same complexity as that arriving at the automaton from the environment. That is, some transformation and reduction of the stimulus information will be necessary before the control center can digest the information and make an appropriate decision. Thus an automaton will contain units whose various functions are transduction, encoding, transmission, decoding, perception, as well as general data reduction of the information present both in the environment and in transit in the automaton.

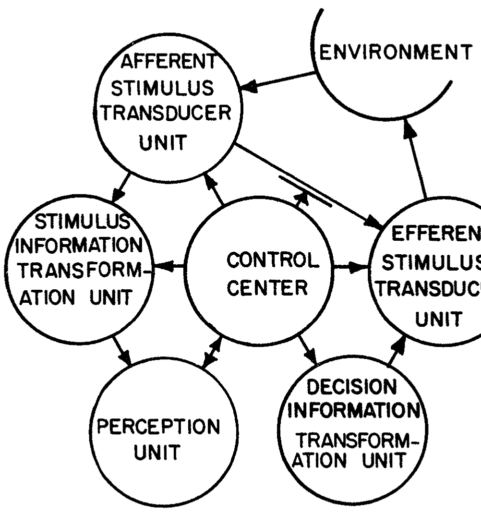

ENVIRONMENT
AFFERENT STIMULUS TRANSDUCER UNIT
STIMULUS INFORMATION TRANSFORMATION UNIT
CONTROL CENTER
EFFERENT STIMULUS TRANSDUCER UNIT
PERCEPTION UNIT
DECISION INFORMATION TRANSFORMATION UNIT

Figure 1. Simple Automaton

Figure 1 illustrates a conceptually simple automaton. The information from the environment must first be processed by a stimulus transducer--i.e., a sensory element--after which it undergoes various transformations and reductions so as to be in suitable form for perception and control center assimilation. Upon assimilation of the information by the control center working in conjunction with the perception unit, a decision as to appropriate action, if any, is made by the control center and the units of the automaton are instructed to act in some manner conducive to the goal of the automaton. Ultimately, the output of the automaton will appear as some action or information in its environment, so a final transduction must be made. Therefore, after the control center makes a decision, its decision information must be transformed in some manner and then finally a transduction to the environment occurs.

For simple automata with limited special purpose behavior, the scheme shown in Figure 1 seems to be adequate. When the automata requirements are more complex, the question of reliability and economy of data processing may dictate some additional units being placed at various places along the information path. For example, is it more expedient to transfer through to the perception unit nearest the control center as much of the information as possible with the contention that more reliability of stimulus classification can then be made or is it more expedient to reduce and classify the stimulus data as near to the afferent stimulus transducer as possible and perhaps depend upon several different parallel information paths to the perception unit near the control center for reliability of classification? Taking a hint from the functioning of various biological organisms, the author believes the latter method to be most expedient. In fact, perhaps in addition to parallel information transmission paths between stimulus transducer unit and perception unit, different information about events in the environment should be obtained through different sensory modes. This means that instead of only redundancy of information in the coding process, redundancy of source and form of information may be desirable. The result would be redundancy of function as well as redundancy of information through coding.

Once the concept of reducing information as near its source in the automaton as possible is accepted, the concept of "property filtering" as conceived at the University of Illinois Biological Computer Laboratory seems applicable[1]. Property filtering as used here is to be understood as information processing networks arranged in parallel computation channels which extract from the set of input data present certain particular subsets of data which characterize the input data in some manner. The properties which filter through the computational networks and are eventually transmitted to the perception unit near the control center are those properties which are pertinent to the operational goals of the automaton. Perhaps initially the property filters of an automaton would be structurally determined by the designer of the automaton, but it is not inconceivable that the property filter networks could be constructed using any of several different types of "synthetic neurons" as designed by many different laboratories[2], the resultant filters being adaptive with respect to time and function. To better illustrate the concept of property filtering some practical and theoretical filters will now be described.

2. The Numa-Rete

The numa-rete is a very clever form of property filter which was conceived and constructed by P. Weston of the University of Illinois Biological Computer Laboratory[1]. The purpose of this filter is to determine the "n-ness" of its environment which consists of dark objects on a light field, the "n-ness" being the number of such objects which appear simultaneously upon the sensory layer--i.e. retina--of the device. Simply, the "numa-rete" counts the number of distinct dark shadows which are present on its retina when an interrogation button is activated, the count appearing on a numerical display panel.

The operation and construction of the numa-rete can be understood by referring to Figure 2. The sensory layer of the device is a planar square array of photo cells upon which the shadows or the objects themselves are placed. The computation layer consists of a similar square array of active elements, each connected to a photo cell in the sensory layer in an absolutely inhibitory manner in a one to one fashion. These active elements are threshold components which may be considered as non-adaptive synthetic neurons. Each active element is connected in a mutually excitatory manner along rows and columns of the computation layer to each of its four immediate neighbors. Completing the connections to each active element is an excitatory interrogation scanner connection and an output connection. Figure 3 shows the connections to a typical active element and the relative weights of each of its connections, including the threshold value of $\theta = 1/2$.

The control and display unit completes the numa-rete. Its purpose is to sequentially interrogate each active element in the computation layer in a manner to be described soon, sum the resulting activity to determine the total object count, and present the result on the numerical display panel.

The operation of the numa-rete is as follows. A number of distinct shadows are cast upon the retina, each shadow's placement and size being within the resolution limits of the device. The operator then presses the interrogation button which sets all the active elements to one bistable state and also starts the interrogation scanner. The active element array is scanned sequentially, element by element, along rows until the scanning pulse interrogates an active element whose corresponding photo cell is in shadow. Since this element is not absolutely inhibited by its photo cell, it is triggered to its other bistable state. This results in any of its neighbors whose corresponding photo cells are also in shadow being triggered to their other bistable state. Neighbors whose photo cells are not in shadow are left unchanged. Therefore all active elements whose photo cells are under the same shadow are forced to change state, regardless of their position in the array. The result of all these active elements changing

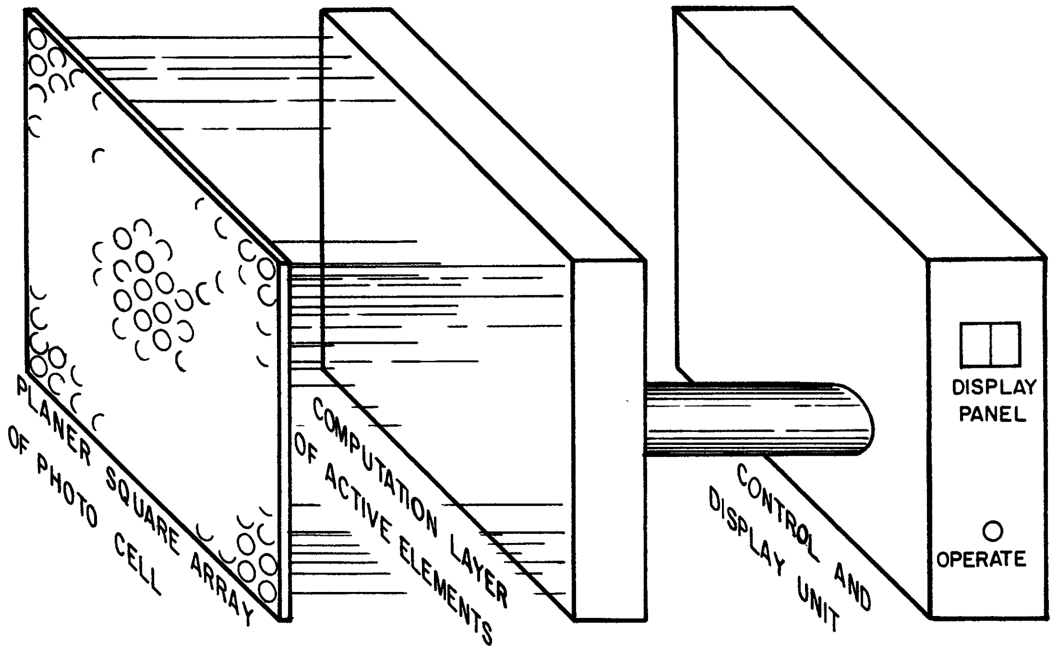

PLANER SQUARE ARRAY OF PHOTO CELL
COMPUTATION LAYER OF ACTIVE ELEMENTS
CONTROL AND DISPLAY UNIT
DISPLAY PANEL
OPERATE

Fig. 2
Numa-rete

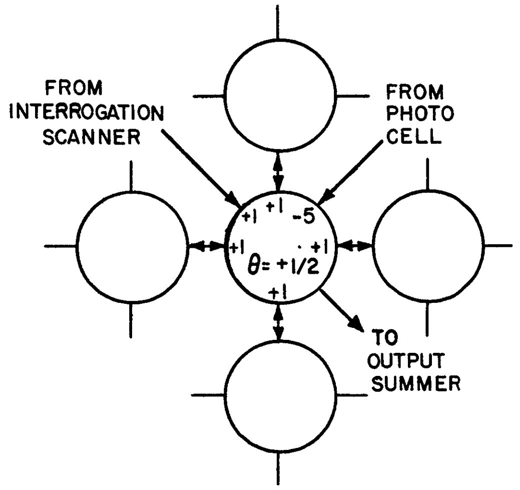


Fig. 3
Typical active element connections in the numa-rete
(Numbers in cell indicate relative weights of connections)

state is an output pulse to the summing unit. In the meantime, the interrogation scanner continues its interrogation of the active elements. Since an active element which has been triggered previously cannot be triggered again until it is reset, only the active elements which are under a different shadow from all previous shadows can be triggered by the interrogation pulse. Therefore, after the scanner has completed interrogating the entire array, the number of output pulses is equal to the number of disconnected shadows appearing on the retina.

The numa-rete has no limitations with respect to shape or position of shadows on its retina other than the obvious ones of size and minimum distance between shadows as determined by the size and placement of the photo cells in the retina. Its operation does depend, however, upon a sequential method of interrogation. For some purposes this might be undesirable and so in an attempt to eliminate this restriction so that parallel operation of the output count could be performed, the method to be described next was devised.

3. Topological Counter

Consider a property filter which would instantaneously determine the total number of disconnected activity classes present at any instant upon some lamina structure; for example, if the lamina is a retina then the property filter would detect the total number of distinct objects focused upon the retina. L. Lofgren of the University of Illinois Biological Computer Laboratory has suggested a unique way to perform this count. This method makes use of the well-known circuit topology equation which relates the number of meshes, nodes, branches, and separate circuits of a network. Reinterpreting the terms mesh, branch, node, and circuit in terms of active elements and various combinations of these active elements within a lamina, the equation in question can be used to determine the number of separate classes of active elements in the afferent lamina in question. Thus:

$$C = M + N - B \qquad (1)$$

where:

C = the number of classes of connected active elements in a specific afferent lamina.
M = the number of particular combinations of active elements, each combination corresponding to an elementary network mesh.
B = the number of particular combinations of active elements, each combination corresponding to an elementary network branch.
N = the number of active elements, each element corresponding to an elementary network node.

Figure 4 illustrates the simplest arrangement of elements and the various combinations of active elements in a lamina which correspond to a node, a branch, and a mesh for use in Equation (1). That is, a node is an active element, a branch is a combination of any two adjacent active elements, and a mesh is a combination of any three triangularly adjacent active elements.

Other than the obvious resolution limitations of size and distance between distinct objects when the lamina in question is a retina, there is one serious fault with this system when used to determine the total number of distinct objects present. These objects must all be "holeless"--i.e. be simply connected--otherwise the total class number, C, is reduced by the quantity, (m - 1), for each m-ordered multiply connected object on the retina. However, when the topological counter is used in conjunction with the numa-rete, this disadvantage becomes an advantage as will be shown later.

As an example of the hardware requirements for such a system, consider a lamina with a structural arrangement like Figure 4 containing (n + 1) rows, each row containing (n + 1) nodes. The resulting maximum total numbers of meshes, nodes, and branches which may occur are:

$$M_{max} = 2n^2 \qquad (2)$$

$$N_{max} = n^2 + 2n + 1 \qquad (3)$$

$$B_{max} = 3n^2 + 2n \qquad (4)$$

Thus it is quite apparent that the number of mesh, node, and branch detectors for parallel computation of all possible lamina stimulus becomes quite large for even a moderately small lamina.

With respect to how the actual calculation of C in Equation (1) can be performed, several different methods may be used. If analog methods are used, then severe restrictions are placed upon the tolerances of the components if the lamina is only reasonably large. Therefore, threshold devices and standard computer techniques may be indicated, if only for the required accuracy. Certainly, synthetic neurons can be used as the threshold devices[2].

Regardless of what methods are used, the final system will have some correspondences to that shown in Figure 5 where a parallel computation method is illustrated for a 16 $\times$ 16 node stimulus lamina. Again the laminae structure possible is visible as it was also in the numa-rete, emphasizing the property filter technique of lamina computation.

As mentioned previously, the numa-rete uses basically a sequential method for its computations

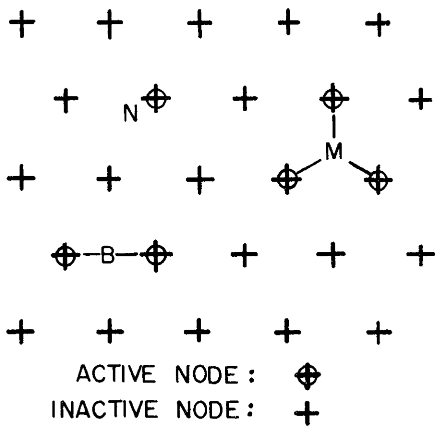


Fig. 4
Typical simple node arrangement for topological counter

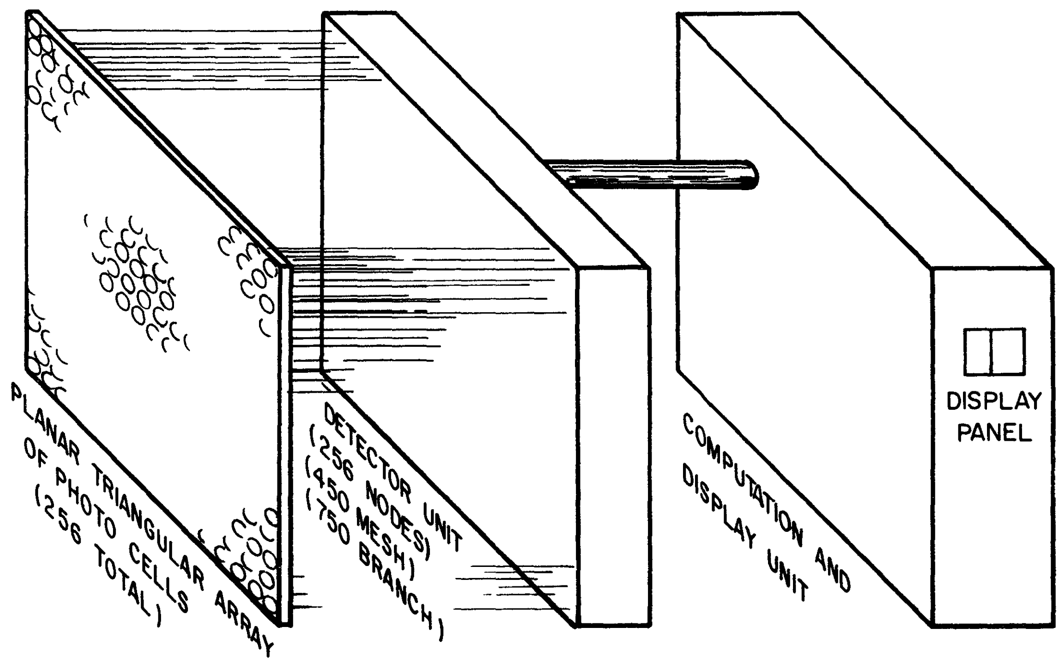

PLANAR TRIANGULAR ARRAY OF PHOTO CELLS (256 TOTAL)
DETECTOR UNIT (256 NODES) (450 MESH) (750 BRANCH)
COMPUTATION AND DISPLAY UNIT
DISPLAY PANEL

Fig. 5
Topological counter with a 16 x 16 node retina

whereas the topological counter just described uses basically a parallel technique. However, the latter could conceivably make use of a modified sequential technique which would be a compromise between the two methods and which might relieve some of the hardware limitations on the topological counter.

4. Theoretical Property Filters

The two property filters, or more properly property detectors in their present use, which have been described are practical solutions to the problem of reducing and classifying information which may be present as afferent stimuli in some lamina. Other reductions and classifications are certainly possible, among them being the ones studied theoretically by A. Inselberg, L. Lofgren, and H. Von Foerster of the University of Illinois Biological Computer Laboratory. A brief description of some of their theoretical property filters follows.

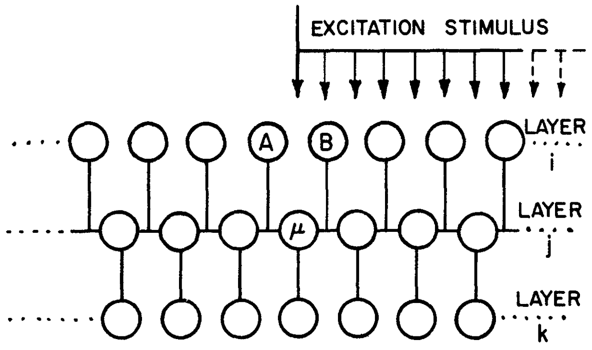


Figure 6. Simple Property Filter

In Figure 6 are illustrated three infinitely extending one-dimensional arrays of elements. Assume layer i is composed of light sensitive elements which can activate the elements in layer j through the connections shown. Assume also that the elements in layer j can each perform some logical functional operation upon the outputs of the particular associated elements in layer i. If, for example, element B and all elements in layer i to the right of element B are excited as illustrated but element A and all elements to the left of element A are not excited, and if each element of layer j performs the logical operation of exclusive "or" on its inputs, then only those elements in layer j which are at the stimulus edge will produce an output. Thus only the element ν, in layer j with output $A\bar{B} \vee \bar{A}B$, will have any output activity for the illustrated instance, all other elements in layer j being inactive. Similarly, a right edge of stimulus would activate the corresponding element in layer j and so the laminae structure considered extracts the property of "edgeness" or contour from the stimulus activity.

If each of the 16 possible logical operations of the elements of layer j upon their inputs are investigated, it will be found that both positive and negative edge detectors, positive and negative right and left edge detectors, positive and negative replicas, and various other characteristics of the original stimulus activity results. In principle, then, various properties of the stimulus can be extracted as a result of these logical operations performed upon the response of one lamina by elements of a succeeding lamina.

This action of one lamina upon another lamina suggests the general concept of some "action-function" between n-dimensional laminae. In fact, if only one lamina is involved, the concept of some "interaction-function" is suggested. The "action-function" would specify how much output activity of any element in lamina i would be passed on to the input of any element in lamina i + n, where $n > o$. For $n = o$, the "action-function" would become an "interaction-function" which would specify how much output activity of any element in lamina i would be passed on to the input of any other element in the same lamina i. If $n < o$, the "action-function" becomes a "feedback-function" which would specify how much output activity of any element in lamina i would be passed on to a preceding lamina. Only "action- and interaction-functions" have thus far been considered.

There is also no conceptual difficulty if the elements are allowed to become indefinitely small and indefinitely close to each other, resulting in a continuum n-dimensional lamina. The resulting stimulus density $\sigma(p)$ and the response density $\rho(p)$ at any point p in the n-dimensional continuum can be defined as follows:[1]

$$\sigma(p) = \lim_{\Delta V^n \to o} \frac{\Delta S(p)}{\Delta V^n} \qquad (5)$$

$$\rho(p) = \lim_{\Delta V^n \to o} \frac{\Delta S(p)}{\Delta V^n} \qquad (6)$$

where:

S(p) = the stimulus or excitation input at point p on some lamina.

R(p) = the response or activity output at point p on some lamina.

V^n = the n-dimensional volume element.

Therefore for an n-dimensional continuum lamina structure a response density function is:

$$\rho_i(p) = \int_j \rho_j(q)F(q,p)ds(q) + \int_i \rho_i(r)G(r,p)ds(r) \quad (7)$$

where:

$\rho_j(q)$	= the response density at point q in lamina j.
F(q,p)	= the action-function between point q in lamina j and point p in lamina i.
ds(q)	= differential n-dimensional volume element in lamina j.
$\int_j$	= integral over all the n-dimensional lamina j.
$\rho_i(r)$	= the response density at point r in lamina i.
G(r,p)	= the interaction-function between point r in lamina i and point p in the same lamina.
ds(r)	= differential n-dimensional volume element in lamina i.
$\int_i$	= integral over all the n-dimensional lamina i.

Using this formalized approach to property filtration, Von Foerster, et al., derive some interesting results for various choices of F(q,p) and G(r,p). For example, if a one-dimensional lamina is excited by a stimulus function, $\sigma(p)$, that can be expanded in a Fourier series with respect to distance along the lamina and if the interaction-function, G(r,p), is chosen as a normal statistical distribution with respect to distance--note that there is no action-function here since only one lamina is being considered--then it can be shown that the lamina in question will perform a "sharpening" of the stimulus function distribution. That is, the higher order terms in the Fourier series expansion are enhanced.

Another interesting example involves two two-dimensional laminae with an action-function, F(q,p), defined as a normal statistical distribution with respect to distance from corresponding points--note that there is no interaction-function here. The result of this analysis, as might be suspected by analogy to the first simple illustrative example, is a contour filter for all two-dimensional response functions of activity from the first lamina, provided the response function can be expanded in a polynomial which contains any distance, x, to a degree no greater than one. This condition can easily be satisfied.

Perhaps the most interesting examples of action- and interaction-functions among laminae are those which involve anisotropy of the functions involved. These result in property filters which may allow certain characteristic information about the geometry or the topology of the lamina activity to be extracted. Of immediate interest are action-functions of the form:

$$F(q,p) = F_1(r,\varphi) = \epsilon^{-(r/r_1)^2} \cos 2\varphi \quad (8)$$

where:

$F_1(r,\varphi)$	= F(q,p) expressed in polar coordinates.
r	= the radial distance from the point p in question to the point corresponding to q in the previous lamina.
φ	= the angle between the radius, r, and some reference line in the lamina in question.
r_1	= arbitrary constant.

If action-functions of this form operate upon the response of some lamina, and if that response is a straight line contour of activity only, then the response of the analyzing lamina will be a function only of the normal distance from the point in question to the straight line contour and of the angle of inclination of the contour with respect to the arbitrarily selected reference line.

The action-functions defined by Equation (8) have certain lines of symmetry with respect to the angle of the radius, r. If more than one anisotropic laminae operate upon the same response function and if the lines of symmetry of each of these laminae are rotated relative to each other, interesting results are obtained when the responses of these computation laminae are summed. In particular, if three parallel anisotropic computation laminae, each differing from the other only in the φ of Equation (8), this difference being $\pi/3$, and the responses of these computation laminae are summed, the result is zero at every point provided the input stimulus is a straight line contour of infinite length. Such a system might be called a "straight line filter".

5. Conclusions

The numa-rete, the topological counter, and the theoretical property filters just described suggest ways in which the afferent stimulus information from an environment can be reduced and classified into responses appropriate to different requirements. Separately, they indeed detect certain limited properties, but in combination, they may be even more useful. For example, the numa-rete and the topological counter operating in parallel can classify an object or character as to its order of connectivity and the numa-rete and the straight line filter in cooperation may be able to dichotomize a group of characters as to whether they are composed of only straight lines or whether they contain curves as well as straight lines.

These concepts lead into the over-all larger property detection area, that of the detection and use of gross object characterizations such as multiple-connectivity ("holeness"), "straight lineness", "angleness", "intersectness", etc. If such property filters as are needed to detect these more gross characteristics of patterns can be designed and constructed, then the way seems open to pattern detection and identification in terms of neighborhood properties only.

TABLE 1

Properties	Printed Digits									
	0	1	2	3	4	5	6	7	8	9
"single holeness"	x				x		x			x
"straight lineness"		x			x			x		
"curveness	x			x			x		x	x
"smoothness"	x	x								
"single intersectness"			x	x				x		

Table 1 illustrates a simple tentative characterization of printed digits by this scheme where the "properties" to be filtered are listed at the left for the corresponding digits across the top. It will be noticed that such characterizations will be independent of size, translation, and rotation, but not entirely from distortion. To make these independent of distortion, topological properties such as the connectivity should be chosen, if possible.

It should be emphasized that the particular characterization illustrated by Table 1 is not yet possible since all the required property filters are not designed and indeed may not be technically possible. However, the concept of property filtering using methods other than standard statistical methods seems to offer some promising possibilities.

Acknowledgment

The author wishes to emphasize that the present article is, in part, a summary of work on data preorganization being done at the Biological Computer Laboratory of the University of Illinois by not only himself, but also by A. Inselberg, L. Lofgren, P. Weston, and G. Zopf, under the direction of H. Von Foerster. The work is sponsored by the Information Systems Branch of the Office of Naval Research.

References

1. Babcock, M. L.; A. Inselberg; L. Lofgren; H. Von Foerster; P. Weston and G. W. Zopf, Jr.: "Some Principles of Preorganization in Self-Organizing Systems," Technical Report No. 2, Contract Nonr 1834(21), Electrical Engineering Research Laboratory, Engineering Experiment Station, University of Illinois, Urbana, Illinois, June 24, 1960, 110 pages.

2. Babcock, M. L.: "Reorganization by Adaptive Automation," Technical Report No. 1, Contract Nonr 1834(21), Electrical Engineering Research Laboratory, Engineering Experiment Station, University of Illinois, Urbana, Illinois, January 15, 1960, 141 pages.

COMBINED ANALOG/DIGITAL COMPUTING ELEMENTS

Hermann Schmid
Link Division, General Precision, Inc.
Binghamton, N.Y.

SUMMARY

This paper describes a number of computing elements whose accuracy is considerably higher than that of comparable analog devices and whose speed of operation is considerably faster than that of comparable digital circuits. These elements employ both analog and digital computing techniques. The input and output signals of these hybrid computing devices are presented by two quantities, the most significant part by a parallel binary number and the least significant part by an analog voltage.

The hybrid multiplier consists of a relatively simple digital multiplier, three D/A converters, a very simple analog multiplier, and a summing and comparison circuit. An accuracy of 1 part in 10^4 and a bandwidth of 1KC is possible with a relatively simple circuit. The accuracy can be expanded theoretically as high as desired at the cost of additional equipment.

The hybrid integrator consists of an accumulating register, a variable amplitude sawtooth generator, a simple analog integrator and a summing and comparison circuit. Its dynamic range and accuracy are limited only by the number of digits used in the R-register.

The hybrid function generator consists of a diode decoding matrix, a diode translation matrix, two D/A converters and a summing and comparison circuit. A repeatability of 1 part in 10^4 and a bandwidth of 1KC is easily possible.

There are no A/D converters, no external storage means, no timing circuits (except clock frequency for integrator), no controls to adjust, and no temperature-sensitive circuits in a system using these hybrid computing elements.

INTRODUCTION

Analog computers are limited to the solution of problems requiring a speed of operation of several hundred cycles, and an accuracy not higher than 0.1% of full scale. In an analog computer each computing element performs, in general, one mathematical operation, and there exists thus a one to one relationship between the complexity of a mathematical problem and the complexity of the computer. We say, the analog computer operates "in parallel".

Digital computers are inherently sequential machines. They perform every mathematical operation through repeated addition or subtraction. The time required to perform a mathematical operation is a function of the time required for one addition and on how many additions are required. Besides, a digital computer performs a large number of mathematical operations in sequence. The larger this number the slower must be the repetition period. Generally speaking, a digital computer is limited to the solution of problems in which the input variables change slowly with respect to time.

The analog and the digital computer are therefore limited to the two distinct fields of computation. When the solution of a certain mathematical problem requires an accuracy higher than 0.1% and a computation speed higher than several hundred cycles neither the analog nor the digital computor is capable of performing this task economically. Hence, a new computer is required for this class of problems.

It is the purpose of this paper to describe a number of combined analog/digital computing elements which have an accuracy-speed product several orders higher than comparable analog circuits. These elements employ both analog and digital computing techniques, and associated with them are hybrid (partly digital, partly analog) input and output signals.

DESCRIPTION OF A HYBRID COMPUTING SYSTEM

An analog computing system is characterized by the fact that all computing elements, such as adders, multipliers, function generators, integrators, etc. accept analog input voltages and produce analog output voltages.

Accordingly in a hybrid computing system all computing elements must accept hybrid input signals and provide hybrid output signals. A hybrid signal is defined as being partly digital and partly analog. The more significant part of the signal is represented by a digital signal X_D in the form of a binary or decimal number or some other digital code. The less significant part is represented by an analog quantity X_A, usually a dc voltage. In this paper the digital signal has been chosen to consist of a three-bit binary number and the analog signal of a dc voltage which varies between 0 and 1.25V, though in many instances it may be more convenient to use other values. If the variables of this hybrid system are related with the variables of a 10V analog system the magnitudes of the 2^0, 2^1, 2^2 bits of the binary number representing the digital portion are 1.25V, 2.5V and 5.0V; respectively.

A magnitude of an input signal X of 6.685V, would therefore be presented as the binary number B 101 and the dc voltage of .435V. As X increases to 7.499V the analog voltage X_A increases to 1.249V. When X becomes 7.501V X_A increases first to 1.250V, then returns to zero, and increases to .001V. As X_A returns to zero the binary number X_D must increase by a magnitude equal to 1.25V, i.e., the 2°digit, in order to maintain the magnitude of the total signal constant. It is important that this changeover will be executed as quickly as possible.

Hybrid computing elements employing DDA (Digital Differential Analyzer) techniques in combination with analog computing techniques have been developed by the National Bureau of Standards[1]. It will be shown later that such a system not only requires more components but its slewing time is limited by its "unit step" digital operation. In addition, if a DDA computer once makes a mistake, as e.g. due to power failure or transients in the system, it would carry that mistake, until the computation is restarted from the initial conditions.

The hybrid computing system described employs only pure digital computing techniques (in addition to the analog techniques). Since all digital operations are carried out in parallel its slewing rate is considerably better. In addition, any momentary failure of the input signals or the equipment has only a momentary effect on the output signal.

From the generalized form of a hybrid computing element in Fig. 1 it can be seen that all inputs and the output consist of a digital part and an analog part. The digital input signal X_D is accepted by a digital computing circuit, while the analog input signal X_A is accepted by an analog computing circuit. There is also one (or more) computing circuit(s) which accept both analog and digital inputs. Since the digital output must have as many digits as the input, the additional digits must be converted back into an analog voltage. The outputs from the D/A converter, the Dig./Anal. computing circuit, and the analog computing circuit are summed in the analog adder. If the analog sum is larger than a certain threshold voltage the comparison circuit subtracts the threshold voltage from the analog sum and provides a "carry" signal which is added to the digital signal. This augmented digital signal is then the digital portion of the output signal. The output from the analog adder is the analog portion of the output signal.

With the number of blocks shown in the diagram of Fig. 1 a hybrid computing element seems to be a complicated and costly device and one begins to wonder whether the increase in cost and complexity justifies the improvement in performance. However, it will be shown that most of these blocks consist of relatively simple electronic circuits.

THE HYBRID MULTIPLIER

One of the best examples on how simple and how cheaply a hybrid computing element can be built is the hybrid multiplier. The circuit described here differs from the NBS version in that the digital input signals are binary numbers and not pulse rates. This eliminates the need for the X and Y-registers, in which the input variables are integrated. Without this integration considerable higher bandwidth is possible at the same clock frequency. Further, instead of the common R-register, which allows multiplication only by a unity step, the device to be described employs a parallel binary multiplier[2].

The operation of the hybrid multiplier shown in Fig. 2 is based on the multiplication of two sums, namely,

$$XY = (X_D+X_A) \times (Y_D+Y_A) = X_DY_D + X_DY_A + X_AY_D + X_AY_A \qquad (1)$$

From the equation above it can be seen that the product XY is now expressed as the sum of four products. At first this may not seem to be a very elegant solution but it will soon become apparent that the circuits required for the individual multiplier elements are relatively simple and that conventional circuitry can be used. From the block diagram in Fig. 2 it can be seen that the first multiplier element is a simultaneous binary multiplier, described in the literature(2). The second and the third multiplier elements accept one digital and one analog signal and thus digital to analog (D/A) converters can be used. The fourth multiplier element can be a simple, one-quadrant analog multiplier. The outputs from all but the digital multiplier element are in analog form and must be added in a conventional analog summing device. The number of digits at the output of the digital multiplier is the sum of the digits of the two input signals. In order that the form of the output signal is compatible with the form of the input signals the output signal should have the same number of digits as the input signal. This can be easily accomplished when the output signal is scaled, i.e., when the binary point is shifted and when the least significant digits are converted back into an analog signal. The latter conversion is achieved by means of the D/A converter #3. The scaling of the output signal will be explained in greater detail on hand of the specific example below.

The real advantage of this multiplier can be appreciated only if the magnitudes of the accuracies required from each multiplier element are described. To simplify this description it is useful to make the assumption, that both input and output signals vary in magnitude between the equivalent of 0 and ±10V.

Before describing the scaling operation in the multiplier for the binary system it is convenient to describe it first for the decimal system. Assume therefore further that the digital multiplier accepts one decimal digit on each input and provides two decimal digits at the output,and that the analog portion of the variable is represented by a DC voltage varying between 0 and 1.00V.If we desire now to multiply, e.g. 9.2 x 8.6 the output from the first multiplier should be 9 x 8 = 72; from the second multiplier the output should be 9 x .6 = 5.4, from the third multiplier the output should be 8 x .2 = 1.6, and from the last multiplier the output should be .2 x .6 = 0.12. When summed this gives the correct value of 79.12. In order to represent the output signal with only one digit it is necessary to scale it down, i.e., to move the decimal point one place to the left and convert everything on the right of the decimal point into an analog signal. When this analog signal is added to the analog sum derived previously and the whole sum is then attenuated by a factor of ten we have a total analog output signal of (5.4 + 1.6 + 0.12 + 2.0) x 0.1 = .912. Together with the single decimal output (7) from the digital multiplier the total output is then 7 + .912 = 7.912, which is the product XY/10.

The magnitudes of the output signals of each of the multipliers in the example above is inversely proportional to the accuracies required from the various multiplier elements. If it is, e.g., desired that the overall multiplier be accurate to 0.01%, then the D/A converter/multipliers must perform to an accuracy of only 0.1% and the analog multiplier must be accurate only to 1%.

Returning now to the actual multiplier, where a variable is represented by a three-digit binary number and a dc voltage varying between 0 and 1.25V, and using the same numbers as in the example above, gives X = 9.2V = 8.75V + 0.45V = B111 + 0.45V and Y = 8,6V = 7.5V + 1.1V = B110 + 1.1V. The product XY then becomes (B111 + 0.45) x (B110 + 1.1V) = B101010 + (8.75 x 1.1V) +(7.5 x 0.45V) + (1.1V x 0.45V) = B 101010 + 13.495V which is, for B 100000 equal to 50V, and B 010000 = 25V, etc., equal to 65.625V + 13.495V = 79.12V. Comparing this with the results obtained above for the decimal system, it can be seen that it is the same result. Proper scaling now requires that the six digit binary number B 101010 is reduced to B101000 and that the rest, B010, is converted back to analog. The voltage equivalent of B 010 is in our example 3.125V and thus the total analog voltage increases to 16.620V. If now the binary point is shifted three places to the left and if the analog voltage is reduced by a factor of 10 we obtain B 101 and 1.662 V which is by our definition 6.25V + 1.662V = 7.912 V, the desired product XY divided by 10.

However, the proper code for 7.912V would be B110 + 0.412 and not B 101 + 1.662V, because by definition the analog voltage can vary only between 0 and 1.25V. It can be seen that if 1.25V is subtracted from 1.662V the remainder is exactly 0.412, and that V 101 is just one binary digit smaller than B110. The "carry" operation, so familiar in digital computers, must also be performed here. This requires that whenever the summed analog voltage is larger than 1.25V, a "one" must be added to the binary number representing the more significant part of the output variable, and 1.25V must be subtracted from the summed analog voltage representing the least significant part of the output variable. The comparison circuit in the block diagram of Fig. 2 performs this comparison and switching operation. A carry signal originating at the comparison circuit is sent to the appropriate adder circuit in the digital multiplier element. It also provides the 1.25V which must be subtracted from the summed analog voltage.

The full advantage of this multiplier will become obvious only when the circuit simplicity of the individual multiplier elements is shown.

The Parallel Binary Multiplier[2] in fig. 3 accepts two input signals with three binary digits each and provides a six digit output. It performs the same algebraic operation as a human operator, multiplying two binary numbers. Binary multiplication is achieved by adding the digits in each column of the array of binary numbers, which is obtained by writing the multiplicand once for every "one" in the multiplier, but always shifted one digit to the left for each increasing digit of the multiplier.

In the circuit of Fig. 3 the "writing" is done by producing coincidences between the specific digits of the multiplicand and that of the multiplier. Only when coincidence occurs the "and" gate produces a "one". The "ones" in each column are summed by conventional adding circuits. The number of components required by parallel binary multiplier rises sharply with the number of digits used at the inputs. With three digits for both X_D and Y_D, the circuit required only 9 gates (2 diodes each) and 9 half adders; if the two inputs have 4 digits each, 16 gates and 16 half adders are needed. In general the circuit requires n^2 gates and n^2 half adders, some buffers (emitter followers) and some additional gating between the half adders to permit full adder operation.

The Digital to Analog (D/A) Converter/Multiplier. For a three-digit input the D/A converter/multiplier in Figure 4 comprises only three complementary transistor voltage switches and three binary-weighted precision resistors. One side of all switches is always connected to ground and the other to the analog input variable X_A or Y_A. When the switches are energized by the control lines of the digital input variable the current flowing to the summing point through the three resistors is proportional to the product X_A Y_D or X_D Y_A.

The Analog Multiplier. It has been mentioned before that the accuracy required from the analog multiplier needs to be only approximately ±1% of full scale in a system with a total accuracy of ± 0.01%, and with 3-digit binary numbers.

An extremely simple version of such a multiplier is shown in Figure 5. Its operation is based on the triangular wave integration principle[5].

A triangular wave is biased to the level of the first input variable X. This biased waveform is then clipped at +Y and -Y, the second multiplication variable, to produce a trapezoidal wave train. When this wave train is subjected to low-pass filtering the resultant output voltage is proportional to the product XY.

Biasing of the triangular wave is obtained by returning the secondary of transformer T-1 to the potential representing X_A. Clipping of the triangular wave is achieved by feeding the triangular wave to the bases of a complementary transistor voltage switch[6]. This switch consists of one pnp and one npn transistors, connected at the emitters. The collector of the pnp transistor is connected to -Y, the collector of the NPN transistor to +Y. The voltage at the emitters of these two transistors is then the biased triangular wave accurately limited to +Y and -Y. Low-pass filtering is done by cascaded RC stages.

The Summing and Comparison Circuit. For ease of understanding, the summing and comparison circuits have been shown in Figure 2 as two blocks, in realty, however, they are combined into one circuit. In essence, this circuit is a simplified version of the partial A/D converter[3] with only one binary digit.

The single flip-flop shown in Figure 6 is supposed to be set when V_B, the output voltage of the d-c amplifier, becomes more negative than 1.25V, and it is supposed to reset when V_B becomes more positive than ground. The output of the flip-flop is used to provide the "CARRY" signal to the adders in the digital multiplier and to energize the transistor voltage switch, which connects - 1.25V to the summing point.

Overall Performance of the Multiplier. The static accuracy of this multiplier is limited only by the complexity and finally by the cost of the circuit, in particular, of the digital multiplier. Theoretically, as in any digital computer, this multiplier can be made as accurate as desired, by using more digits for the more significant part of the variable. Practically, however, it may be most economical to operate the hybrid multiplier with an accuracy of 1 part in 10^4 to 10^5.

In contrast to the digital computer the resolution of these hybrid circuits is very high, due to the analog portion of the variable, and is limited only by the steps occuring during switchover.

The dynamic range of this multiplier is determined by the noise in the circuit, which consists mainly of the drift in the d-c amplifier, the voltage drop across the transistor voltage switches and the integrated effect of the transients occuring during the switchover. In general this noise is below lmV and thus in a 10V full scale system, the dynamic range would be 10,000 or higher.

The frequency response of such a hybrid multiplier would also be relatively high since the bandwidth of the individual multiplier elements can be made high. It is expected that this multiplier should be able to handle frequencies above 1KC.

THE HYBRID INTEGRATOR

The hybrid integrator described in this paper differs from the NBS version because it employs pure digital techniques instead of the DDA techniques. This eliminates the need for an additional register in which the input sighal is integrated. Without this integration higher bandwidth is possible. In order to assure that the output approached a continuous function, the NBS integrator employs a resettable integrator, whereas the integrator to be described uses a variable-amplitude sawtooth generator. The replacement of the resettable integrator by a variable-amplitude sawtooth generator is a major circuit simplification.

If the independent variable X, which is represented by a digital value X_D and by an analog value X_A, is integrated with respect to time, the integral becomes

$$Y = \frac{1}{T}\int_0^t (X_D + X_A)\,dt = \frac{1}{T}\int_0^t X_D\,dt + \frac{1}{T}\int_0^t X_A\,dt \qquad (2)$$

Further, if the time is divided into equal intervals of duration Δt and if the digital quantity is permitted to change its value only at the beginning of Δt, then the integral can be expressed as

$$Y = \frac{1}{T}\left\{\sum_{i=1}^{n-1} X_{Di}\,\Delta t + X_{Dn}\left[t - (n-1)\Delta t\right] + \int_0^t X_A\,dt\right\} \qquad (3)$$

Where X_{Di} is the value of X_D at the i-th interval of Δt, and X_{Dn} is its value in the n-th interval.

In Fig. 7 the integral is represented as the area under the curve X(t) from the time t = 0 to an arbitrary t. In the graph and in the equations above it has been assumed that the initial value of Y is zero. The three terms within the brackets of equation (3) correspond to the three areas depicted in Fig. 7. Area 1 is the integral of the digital part of X between t = 0 and t = $(n-1)\Delta t$; area 2 is the integral of X_D between $(n-1)\Delta t$ and t; area 3 is the integral of the analog part of X between t = 0 and t.

The digital parts of the input and output signals of the Link hybrid computing elements are parallel binary signals, and therefore no input register is required. The Link hybrid integrator in Fig. 8 consists of three parts which produce the three signals that correspond to the three areas in Fig. 7.

In the digital part of the integrator X_D is added into the R-register at the beginning of each Δt. The contents of the R-register is a n-digit binary number representing area 1, where n is a function of the dynamic range desired. k of the n-digits (in Fig. 8 the constant k is assumed to be 3) represent directly the digital output of the integrator. The remaining n-k digits are converted back into the analog voltage V_1.

In the hybrid part of the integrator a variable amplitude sawtooth generator provides the voltage V_2 which is proportional, to both X_D and to t. The variable amplitude sawtooth generator can, but need not be a D/A converter and a resettable integrator, if X_D is kept constant during the time interval Δt. The output from the resettable NBS integrator is nothing but a sawtooth with variable amplitude and constant period Δt. It is therefore suggested for the Link hybrid integrator that only a D/A converter be used and that the reference voltage V_R be replaced by a sawtooth wave with a constant amplitude and period. The output from the D/A converter will then be also a sawtooth wave with an amplitude proportional to X_D. The resettable integrator in the NBS circuit is a rather complicated device which consists of an analog integrating amplifier and a pulse switching circuit to discharge the integrating capacitor in as short a time as possible. The replacement of it simplifies the integrator circuit significantly. The reference sawtooth used in the Link integrator needs to be generated only once for a number of integrators.

In the third part the analog portion X_a of the input variable X is integrated into the voltage V_3 by a conventional integrating amplifier.

The three voltages V_1, V_2, V_3 are summed to produce the analog output voltage of the integrator V_{YA}, which is also compared with an upper or lower threshold voltage $\pm V_{th}$. When $V_{YA} \gg + V_{th}$ the comparator produces a "carry" signal which adds "1" to the contents of the R-register and subtracts V_{th} from V_{YA}; when $V_{YA} \leqslant - V_{th}$ the carry is removed from the R-register.

The R-register accepts the X_D input lines and adds the value of X_D to the binary number stored already in the register. There are several methods of achieving this task. Fig. 9 illustrates a straight-forward version of such a circuit, which consists of a parallel-binary adder and a n-digit binary counter. The parallel adder sums X_D with the outputs from the last three digits of the counter. Its output is connected to the inputs of the least significant stages of the counter in order to "set" or "reset" the flip-flops. To assure that the register output is changing only at the beginning of each t the digit lines of X_D are gated with the clock frequency by means of conventional "And" circuitry.

The Variable Amplitude Sawtooth Generator must generate a sawtooth wave with constant period and an amplitude proportional to X_D. As mentioned before this can be accomplished by connecting a sawtooth wave with constant period and constant amplitude into a conventional D/A converter as shown in Fig. 4.

The D/A converter attenuates the reference sawtooth wave accurately to an amplitude which is directly proportional to X_D. The number of stages required for the D/A converter is identical with the number of digits used for X_D.

Since there exists a multitude of circuits which generate a sawtooth wave with constant period and constant amplitude this circuit will not be described here.

The Analog Integrator is an operational amplifier with an integrator capacitor in its feedback path, however, it can be just as well only a simple RC integrator, if more digits are used for X_D.

The Summing and Comparison Circuit must add the input voltages V_1 to V_3 and compare this sum with a positive and with a negative threshold voltage V_{th} in exactly the same fashion as was described for the hybrid multiplier. The "carry" signal produced by the comparison circuit must be added to the contents of the R-register.

THE LINEAR-SEGMENT HYBRID FUNCTION GENERATOR

Function generators in which the output signal is related to the input signal in any arbitrary fashion usually employ linear-segment approximation principles. In order to design such a function generator the designer must know the values of the function at certain fixed points of the input variable, and the slope of the function between any two adjacent fixed points.

In present-art linear-segment diode function generators, a reverse-biased diode becomes conductive when the input signal becomes equal to or larger than a certain fixed value, which is referred to as breakpoint. The amount of reverse bias on the diode is equal to the value of the function at the breakpoint. Each diode that has been made conducting decreases the slope of the function for a particular segment. It can therefore be said that the linear-segment diode function generator stores the value of the function for each breakpoint and the value of the slope between any two adjacent breakpoints. In a 100V computing system the accuracy with which a diode function generator can retrace the linear-segment curve is of the order of 0.1%. This requires that the circuit has been carefully set up, which is generally a very tedious and time-consuming job. When the diode function generator must operate in a 10V computing system its repeatability is considerably worse, due to the inherent noise or threshold level of the diodes.

The function generator to be described in this paper employs also, the linear-segment principle, but it stores as parallel binary numbers the value of the function and the value of the slope. In contrast to the biased-diode function generator in which all operations are performed by analog computing elements, the function generator to be described performs only the linear operation of summing with a dc amplifier; all other operations are performed with digital or hybrid computing elements.

The combination of analog and digital computing techniques lends itself very well to the generation of arbitrary functions of one or two variables . Such hybrid function generators exhibit unusually high repeatability combined with a relatively high speed of operation.

Compared with the biased-diode function generator the following advantages can be listed for the hybrid function generator:

1. The repeatability is at least one order higher.
2. The speed of operation is the same.
3. The circuit is largely insensitive to temperature variations, since diodes and transistors are used only as digital switching elements.
4. The generator has no controls to adjust and no biases to be set.
5. The function generator can be set up or changed from one function to another by inserting pins in a patchboard or by inserting a punched card. Which of these methods is used depends entirely on the intended application.

A linear-segment function generator can only approach the curve $Y = f(x)$ in Fig. 10. The eight breakpoints P_0, P_1, P_2, etc. on the curve have the abscissae x_0, x_1, x_2, etc. and the ordinates $f(x_0)$, $f(x_1)$, $f(x_2)$ etc., respectively. The eight equal-spaced values of x correspond to the eight values of the three-digit binary number representing x_D. The slope between any two breakpoints is

$$\frac{f(x_{i+1}) - f(x_i)}{x_{(i+1)} - x_i} = \frac{\Delta f(x_i)}{\Delta x_i} = \Delta f(x_i) \text{ when } \Delta x_i = 1 \qquad (4)$$

In order to find the value of $Y=f(x)$ for a value of x that lies between two adjacent values of x_i, i.e., between x_i and x_{i+1} an increment must be added to the value of the function at x_i. This increment is the product of the slope of the function between these two breakpoints and the increment of x, defined generally as x' or as x_A in the hybrid computing system. The value of the function for any magnitude of the input variable can be expressed for any linear-segment approximation as

$$f(x) = f(x_i) + x'\Delta f(x_i) \qquad (5)$$

Every computing element described in this paper is required to accept hybrid input signals and to provide hybrid output signals. The final version of the hybrid function generator will also satisfy this requirement. However, it is convenient to describe at first the function generator with hybrid input and analog output as shown in Fig. 11. A function generator of this kind, but in combination with a partial A/D converter was disclosed previously[3]. At that time it was desired to have a function generator with analog input and analog output. This necessitated the use of a partial A/D converter, which reduced the dynamic performance of the function generator. In a computing system where all variables are represented by hybrid signals no A/D converter is required and the function generator is thus capable of operating with relatively high input frequencies.

The function generator in Fig. 11 comprises a diode decoding matrix, a diode translation matrix, two digital to analog (D/A) converters and a dc amplifier.

The diode decoding matrix converts the three-digit binary numbers representing the digital portion (x_D) of the input variable x into eight control lines. Only one of these eight control lines is energized at any one time.

Each of these eight control lines is connected to one horizontal bus wire of the diode translation matrix. The diode translation matrix provides two parallel binary numbers as outputs if one of its eight inputs is activated. The k-digit number originating in section #1 represents the value of the function $f(x_D)$ for a specific digital value x_D, the m-digit number originating in section #2 represents the value of the slope $f(x_D)$ in the x_D segment.

Each of the two D/A converters accepts one of the two binary numbers as digital inputs and provides an output which is proportional to the product of the digital signal and the reference potential. The number representing $f(x_D)$ is converted into an analog voltage by D/A converter #1. Its reference voltage is V_R, and its analog output voltage is thus $V_R\ f(x_D)$. The number representing the slope $\Delta f(x_D)$ is converted into an analog voltage by D/A converter #2 to which V_{xA} - the voltage representing the analog portion of the input signal - is connected as reference voltage. The output from the second converter is thus $V_{xA} \times \Delta f(x_D)$.

When the outputs from the two D/A converters are summed in a conventional DC operational amplifier a voltage is obtained the magnitude of which is proportional to the desired value of the function, i.e.

$$V_o(x) = V_R\, f(x_D) + V_{x_A} \Delta f(x_D) \qquad (6)$$

It has therefore been shown that the function generator in Fig. 11 is capable of generating arbitrary functions of one variable. Although the output is an analog voltage there are certain instances where this function generator may be used also in a hybrid computing system. If, however, the function generator must drive another hybrid computing element it must provide an output signal which is also in the hybrid form.

A function generator with hybrid input and hybrid output signals is illustrated in Fig. 12. The diode decoding matrix, the diode translation matrix and D/A converter #2 are identical to those shown in Fig. 11. For any specific x_D input, the two sections of the diode translation matrix provide two binary numbers as outputs; sections #1 provides a k-digit binary number representing $f(x_D)$, while section #2 provides a m-digit binary number representing $\Delta f(x_D)$. D/A converter #2 accepts the number for $\Delta f(x_D)$ and multiplies it with V_{xA}, the voltage representing the analog portion of the input variable, in the same manner as described before, the k-digit binary number for $f(x_D)$ is now split up into the three most-significant digits and into the k-3 least significant digits. The latter digits are connected to D/A converter #1, which converts them into an analog voltage $V_R\, f_1(X_D)$. The outputs from the two D/A converters are summed again by a standard DC operational amplifier.

The output from the DC amplifier constitutes the analog portion V_{Y_A} of the hybrid output signal. This analog voltage is then compared with a threshold voltage in the comparison and switching circuit. If $V_{YA} \geqslant + V_{th}$, a voltage is fed back to the summing point of the dc amplifier to subtract V_{th} from V_{Y_A} and at the same time a digital signal is provided, which indicates that V_{Y_A} has exceeded the threshold voltage. When $V_{Y_A} \leqslant - V_{th}$, the feedback signal and the digital signal are removed.

A parallel binary adder sums the three most significant digits from section #1 of the diode translation matrix with the digital (carry) signal from the comparison circuit. The output from the adder circuit is another three digit parallel binary number which represents the more significant part of the output variable Y.

With this hybrid form of output representation the repeatability and dynamic range of the function generator can be, theoretically, extended as high as desired by using more digits.

Most of the circuitry used in the linear-segment hybrid function generators is conventional and therefore emphasis is put to the general circuit form, to special components used, to the number of subcircuits used and to their interconnections. Only where the circuits differ from the basic conventional design will they be described in detail.

The Diode Decoding Matrix. This matrix is of conventional design and consist of 2^s "and" gates, each having s diodes. For example in Figure 11, and 12, the number s =3. Therefore the required matrix must have 8 "And" gates, each with 3 diodes. Because the outputs from the diode decoding matrix must be able to drive the diode translation matrix they are passed through conventional emitter followers.

The Diode Translation Matrix. This matrix is an array of horizontal and vertical bus wires. There are 2^s horizontal wires which accept as inputs the signals on the s^2 control (or output) lines from the diode decoding matrix. An input signal can exist only at one horizontal wire at a time. For each input the translation matrix must provide as output two parallel binary numbers. For this reason the vertical wires are split up into two sections, each of which has as many vertical wires as there are digits in the binary number. Section number one, which is to represent the value of the function $f(x_D)$, at one of the 2^s breakpoints must have k digits; section #2, which is to represent the slope of the function $\Delta f(x_D)$ between the breakpoints x_i and x_{i+1}, has m digits. The numbers m and k depend on the accuracy desired from the circuit. Each vertical line has therefore a specific binary significance.

In order to have a specific binary number as output from each of the two sections, it is necessary to provide a low impedance path between the particular horizontal line that has been energized and the appropriate vertical lines. If e.g., the number desired is 011000..., only connections to the second and the third most significant digit lines are required, for all the "1" (ones) in the binary number.

The connections required must be such that a low impedance between a horizontal and a vertical line exists only when the particular horizontal line is energized, for all the other unenergized horizontal lines a high impedance must exist.

There are several possible ways in which two lines can be connected with each other, still fulfilling the requirements set forth above. However, if speed of operation and cost must be considered the semiconductor diode seems to be the most favorable component for this job.

The operation of the maxtrix will be understood better by referring to Figure 13 where a circuit is shown, which solves the function $f(x) = x^2$. This particular function has been chosen because it requires only a few diodes in the translation matrix, and because the performance of the function generator is simple to check.

The circuit in Figure 13 is used for the actual breadboard, for which there are 8 control lines and thus 8 horizontal lines in the translation matrix. Since $f(x_D)$ and $\Delta f(x_D)$ are represented by an 11-digit binary number, there are thus 11 vertical lines in each section of the matrix. For convenience these vertical lines are "weighted" from 2^6 for the most significant digit to 2^{-4} for the least significant digit. The input signal to the horizontal lines are +15V when the line is energized, and -15V when the line is not energized. The breakpoints had been chosen as follows: $x_0 = 0$, $x_1 = 1$, $x_2 = 2$, $x_3 = 3$, etc. For these values of x_D the values of the function are $f(x_0) = 0$, $f(x_1) = 1$, $f(x_2) = 4$, $f(x_3) = 9$, etc., and the values of the slope are $\Delta f(x_0) = 1$, $\Delta f(x_1) = 3$, $\Delta f(x_2) = 5$, $\Delta f(x_3) = 7$, etc. The diode translation matrix which can also be looked upon as a memory must therefore be loaded with the binary numbers for the values of $f(x_D)$ and $\Delta f(x_D)$. Since they are all integers the digits 2^{-1} to 2^{-4} are not required. When the maximum number of diodes on one vertical line is eight, one is conducting and seven are cutoff. The ratio between the forward impedance and the cutoff impedance of the individual diodes should be at least 10 times higher than the 7:1 ratio mentioned above, to obtain a signal to noise ratio larger than 10:1. Even the cheapest diodes on the market will fulfill this requirement. The signals on the vertical lines are the outputs of the translation matrix, which must control the switches in the D/A converters. The amplitude of these output signals must thus be ±15V, and the output impedance must be low enough to quarantee a current of 1mA. All diodes on one vertical line form an "OR" gate. In order to provide -15V at the output it is necessary to return the "OR" gate to a negative potential.

The Digital to Analog (D/A Converter) used with the linear-segment hybrid function generator is based on the voltage decoder with ladder network II, which is described by A. Susskind[4]. The voltage generators shown in the above mentioned reference are replaced with complementary transistor voltage switches which connect either zero or the reference potential to the ladder network. The basic characteristic of a ladder network is such that the current contribution from each stage is half the current contribution of the previous or more significant stage.

The number of stages of the ladder circuit is determined by the number of digits used in section #1 or #2 of the diode translation matrix. There is no theoretical limit to the number of stages in the ladder network; a practical limit is the noise level in the system. Because the signal to noise ratio on these transistor switches is higher than 10,000:1, the D/A converter is capable of operating essentially with the same accuracy as the accuracy of the resistors used in the ladder network.

The Summing and Comparison Circuit shown in Fig. 12 for the hybrid function generator is identical in hardware and performance as that described for the hybrid multiplier and illustrated in Fig. 6.

The DC Operational Amplifier used for summing and comparison is operating most of the time as simple summing amplifier with a 1:1 feedback ratio. But, when V_{YA} exceeds the upper or lower threshold voltage the input to this amplifier is changed rapidly and the output is required to follow as fast as possible. Since, however, V_{YA} must be accurate only to approximately 1/10, (for three-digit hybrid signals) the requirements on drift stability and linearity are not extremely high. Summing up the dc amplifier should provide a gain of appr. 10^4 and a bandwidth of 10KC or higher.

The Performance of the Hybrid Function Generator. Both the static and dynamic performance of the hybrid function generator are limited by the performance of the D/A converters, by the summing and comparison circuit and by how many digits are used in the digital portion of the input and output variable.

With the circuit of Fig. 12 a static repeatability of ±0.01% of full scale output has been measured, but better repeatability is possible, even with three digits, since the analog portion of the output signal alone can be made to be accurate to 0.05%.

For any linear-segment function generator repeatability expresses how closely the circuit approximates a given curve. The latter can therefore be used only when reference is made to a specific curve and when the number and position of the breakpoints is defined.

Dynamic performance results are not available yet, but it is expected that the circuit will operate with input frequencies of several kilocycles. The major limitation on bandwidth is placed upon the circuit by the dc amplifier, used for summing and comparison, which is required to change its output signal from the threshold voltage back to zero in as short a time as possible.

ADDITION OF HYBRID SIGNALS

In analog computers the addition of two or more signals can be easily performed by connecting the signals over suitable resistors to the summing point of a dc operational amplifier.

In digital computers the addition of two n-digit binary numbers requires n "full adders" if the operation is to be performed in parallel.

Consequently the addition of two hybrid signals, each consisting of a k-digit binary number and a dc-voltage, requires k "full adders" ($K \ll n$) and a dc operational amplifier. Since the sum of the two analog signals may exceed the full-scale value a comparison circuit is also needed to provide the "carry" to the digital adder.

CONCLUSION

The hybrid computing elements described in this paper, with their expandable accuracy and relatively high speed of operation, are believed to be most useful in the solution of problems requiring an accuracy one or two orders higher than that of comparable analog circuitry and a speed of operation in the kilo cycles region.

A breadboard has been built for each of the hybrid circuits described in order to confirm the basic principle of operation. Although only the hybrid function generator has been subjected to thorough static testing a number of basic problems have come up, which require further attention.

The speed of operation of all hybrid computing elements is largely determined by the speed of the comparison circuit. More specifically it is the DC amplifier used with that circuit, which must change its output from full scale to zero during a transition from analog to digital, and vice versa. If the amplifier output returns to zero <u>after</u> the digital number has changed, then the <u>total</u> output is for a short time too large, if it returns <u>before</u>, the output is too small. When the digital and analog outputs are appropriately combined and displayed this overlapping or ambiguity shows up as a positive or a negative spike, the duration of which is equal to the time difference between the analog and digital operations. Since the frequency of these spikes is high compared to the basic operating frequency it should be possible to filter them out. However, the integrated transients produce a change in the amplitude of the output signal and thus an error, whose magnitude can not be neglected. In closed-loop systems both the unfiltered transients and the discontinuity of output signals during the switchover have adverse effect on the stability characteristic.

Another problem to be analyzed is how to handle variables with both positive and negative polarities. In the multiplier, e.g., it would be most advantageous to represent the input variable in the "signed magnitude" code in order to keep the digital circuit as simple as possible. All circuits involving adders, on the other hand, prefer numbers represented in the "two's" complement" form. Therefore some compromise solution must be found.

A third problem is that of scaling or multiplying by a constant. In analog circuits this is easily achieved by changing the scaling or summing resistors. In hybrid computers it may require as much as a simplified multiplier circuit.

These, and probably many more, problems must be solved before a hybrid computing system with the elements described will become a useful tool.

REFERENCES

1. Skramstad, H.K., "A Combined Analog-Digital Differential Analyzer", 1959 Proceedings of the Eastern Joint Computer Conference.

2. Richards, R.K., "Arithmetic Operations in Digital Computers", D. Van Nostrand Company, New York, 1958.

3. Schmid, H. "High-Speed, High-Accuracy, Linear-Segment Function Generators", Proceedings of the Combined Analog Digital Computer Systems Symposium, Dec. 16/17, 1960, Philadelphia.

4. Susskind, A. "Notes on analog to digital conversion techniques", Technology Press 1957, MIT, Boston, Mass.

5. Meyers, R.H. & Davis, H.B., "Triangular Wave Analog Multiplier", Electronics, August, 1956, pp. 182-185.

6. Schmid, H., "A Transistor Bidirectional Limiter", Semiconductor Products, April 1960, pp. 29-32.

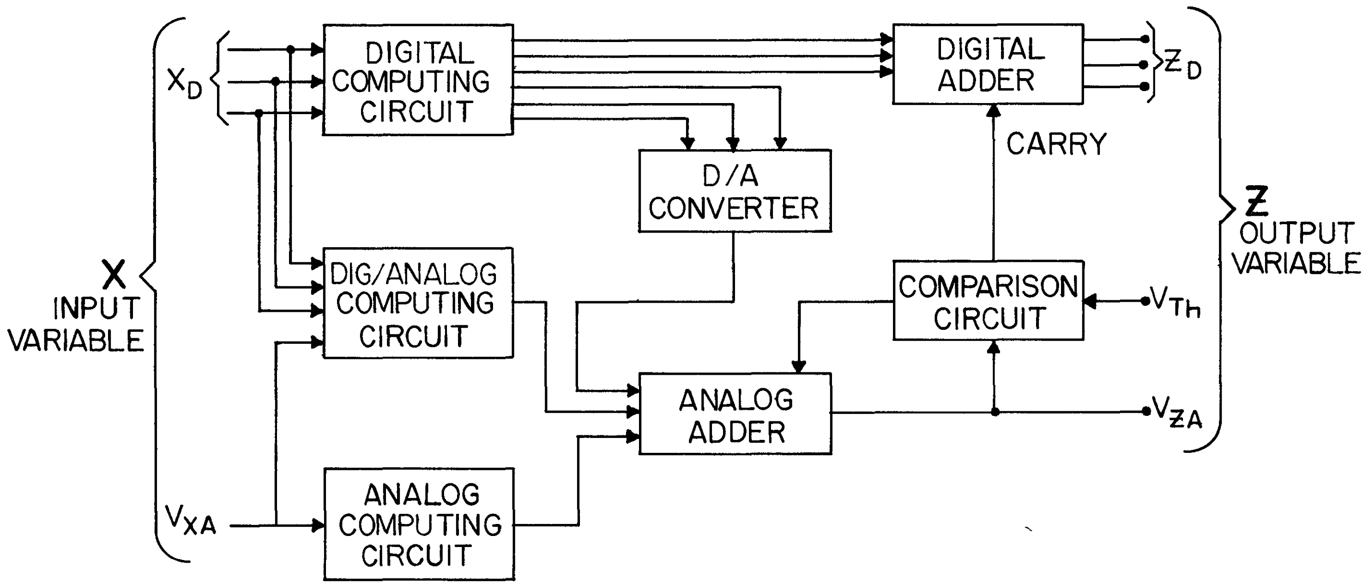


Fig. 1. Generalized form of a hybrid computing element.

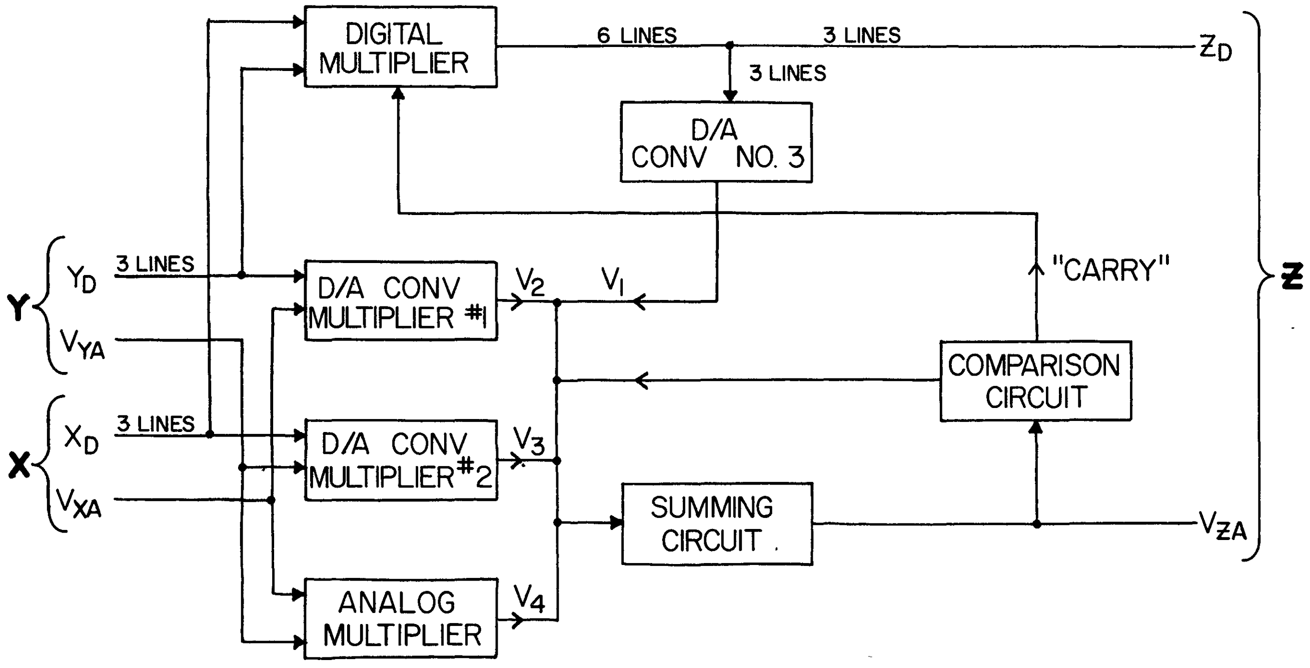


Fig. 2 Block diagram of the hybrid multiplier.

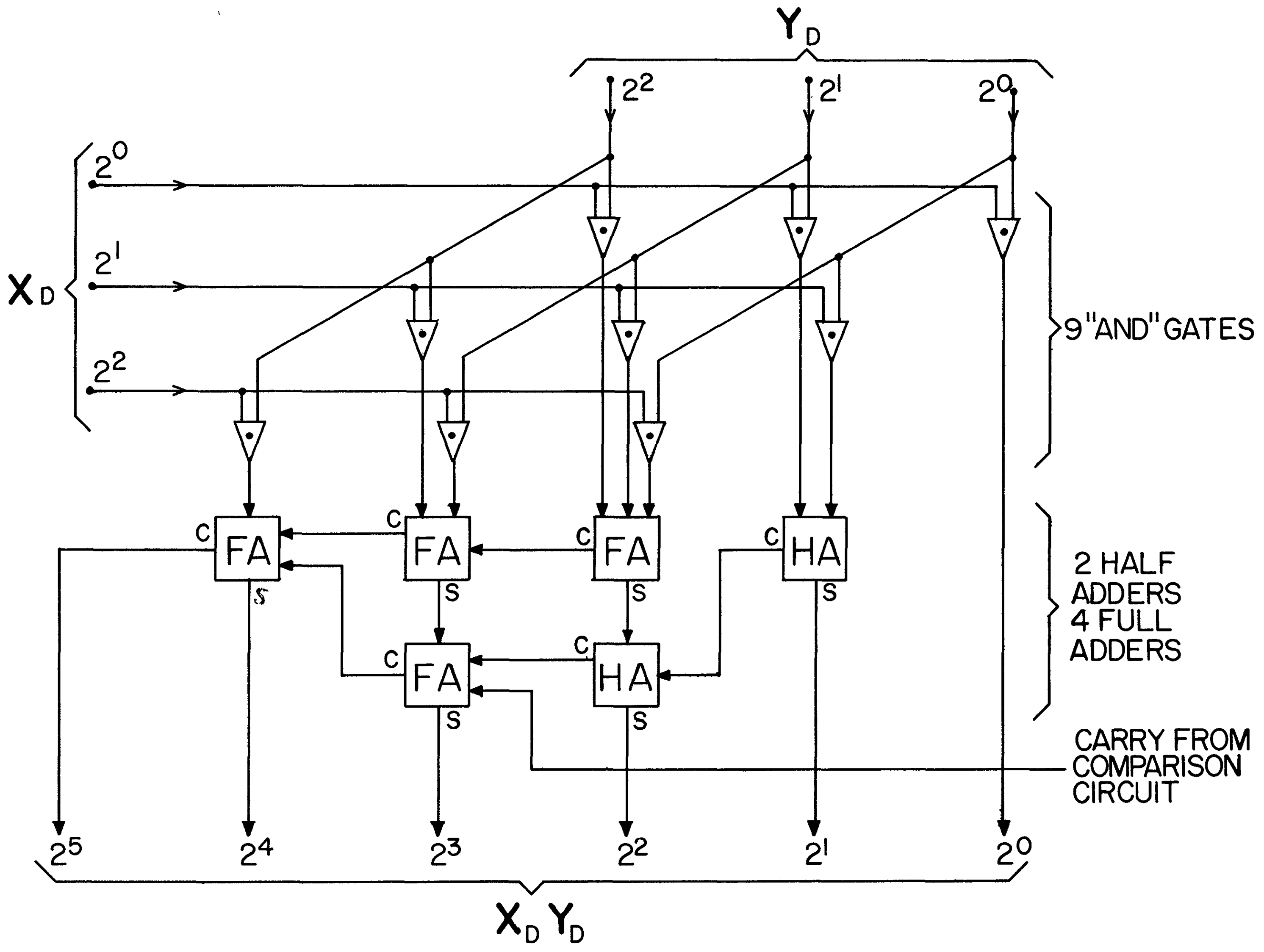


Fig. 3. Parallel binary multiplier for two 3-digit numbers.

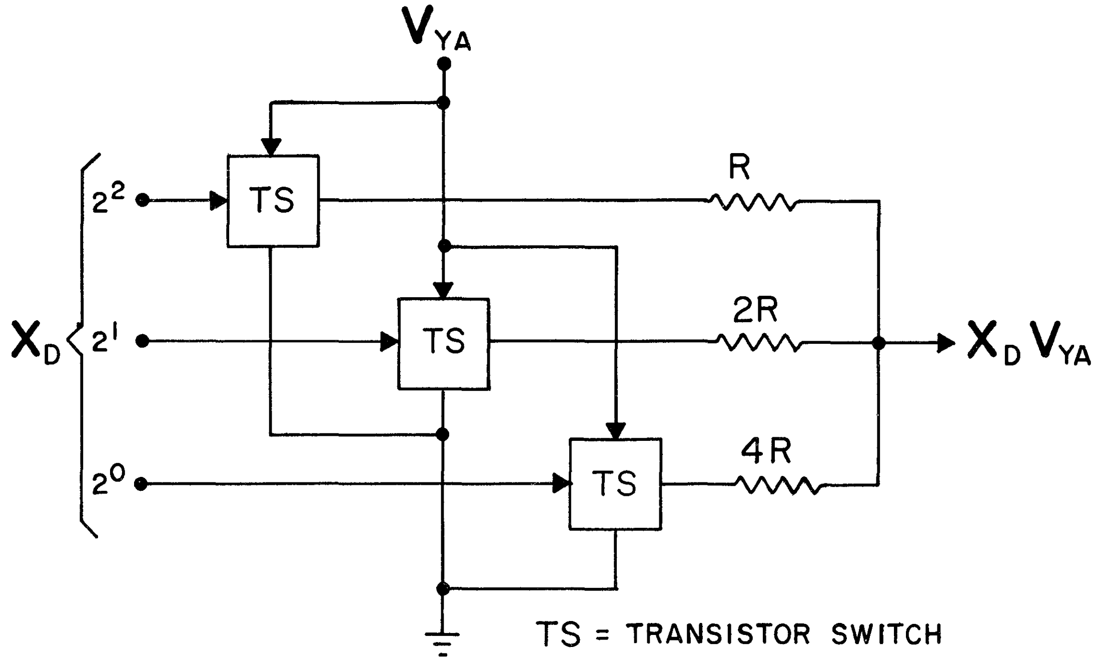


Fig. 4. Three-digit D/A converter/multiplier.

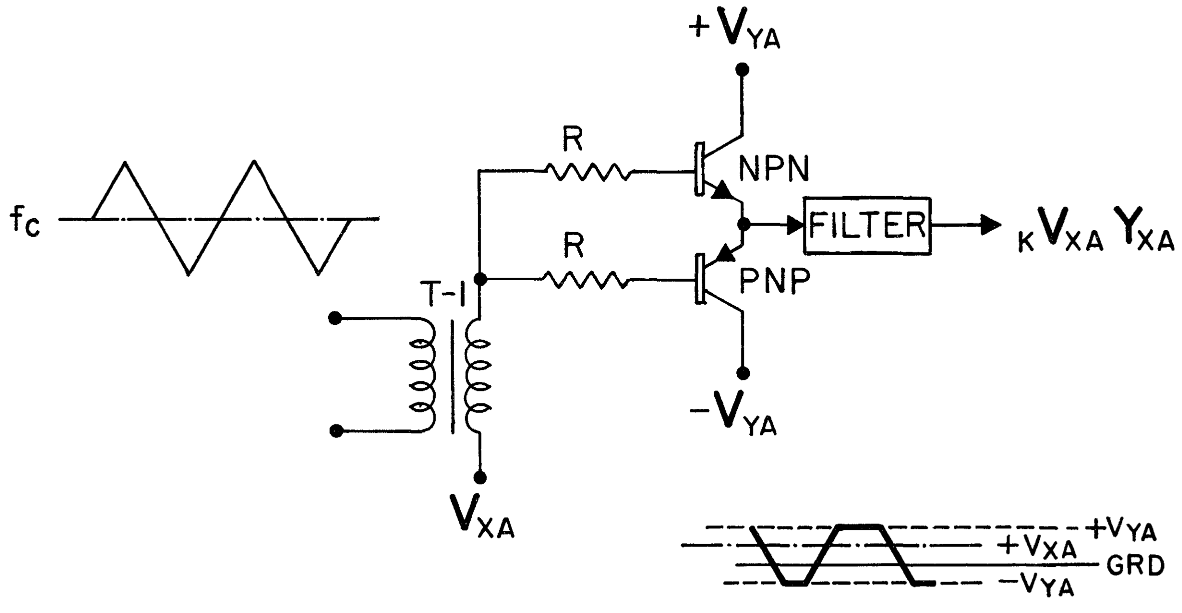


Fig. 5. Two-quadrant analog multiplier.

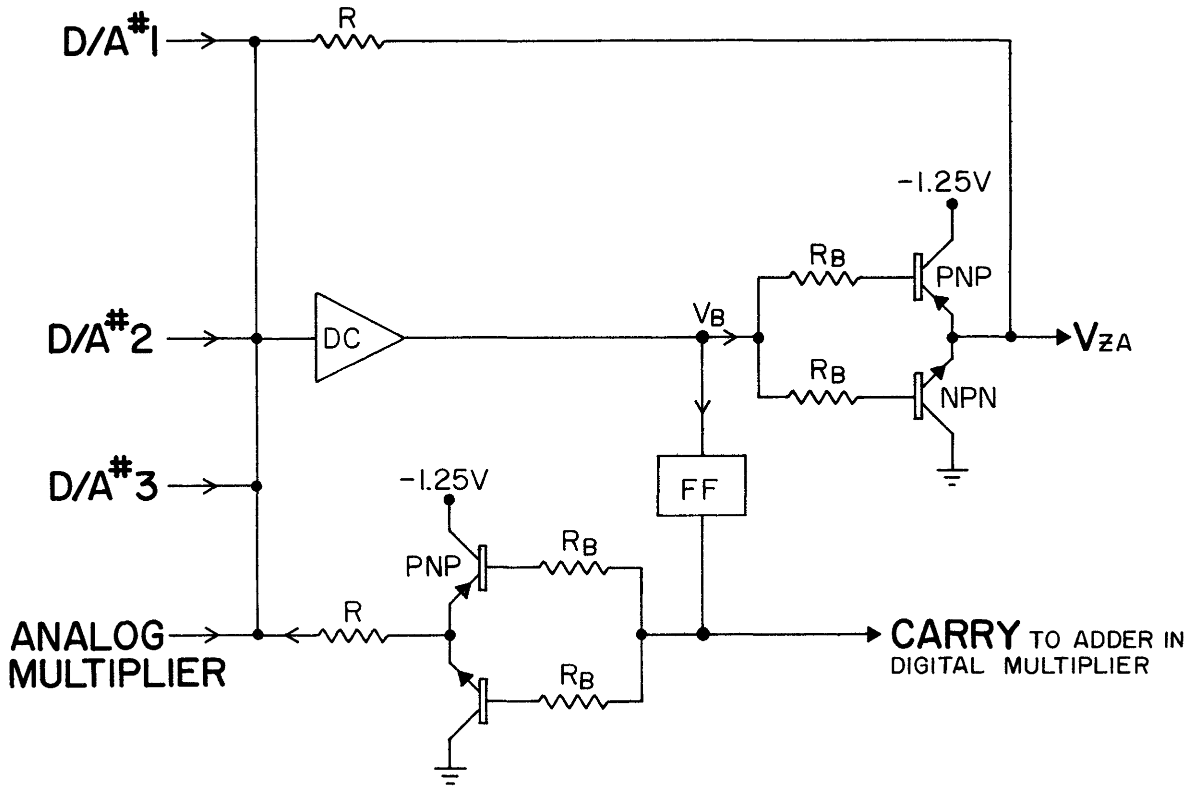


Fig. 6. Summing and comparison circuit.

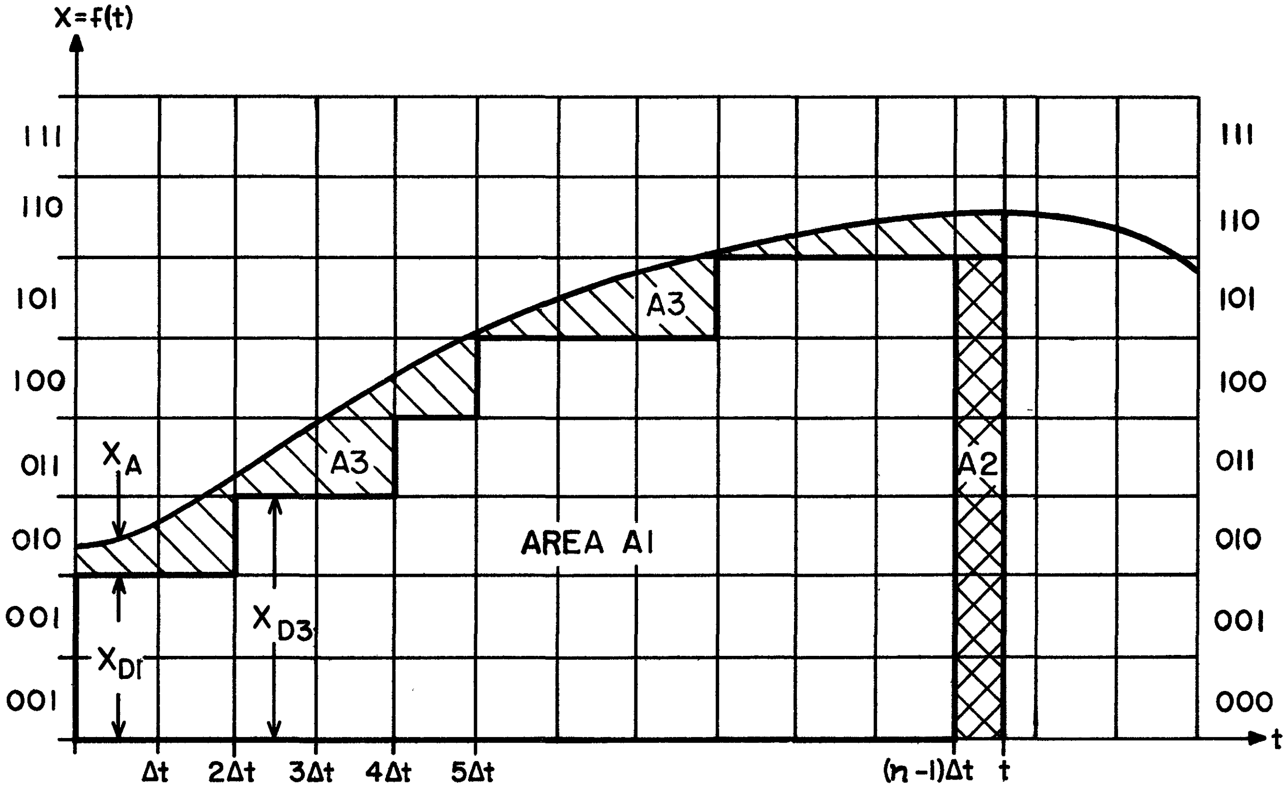


Fig. 7. Hybrid integration illustration.

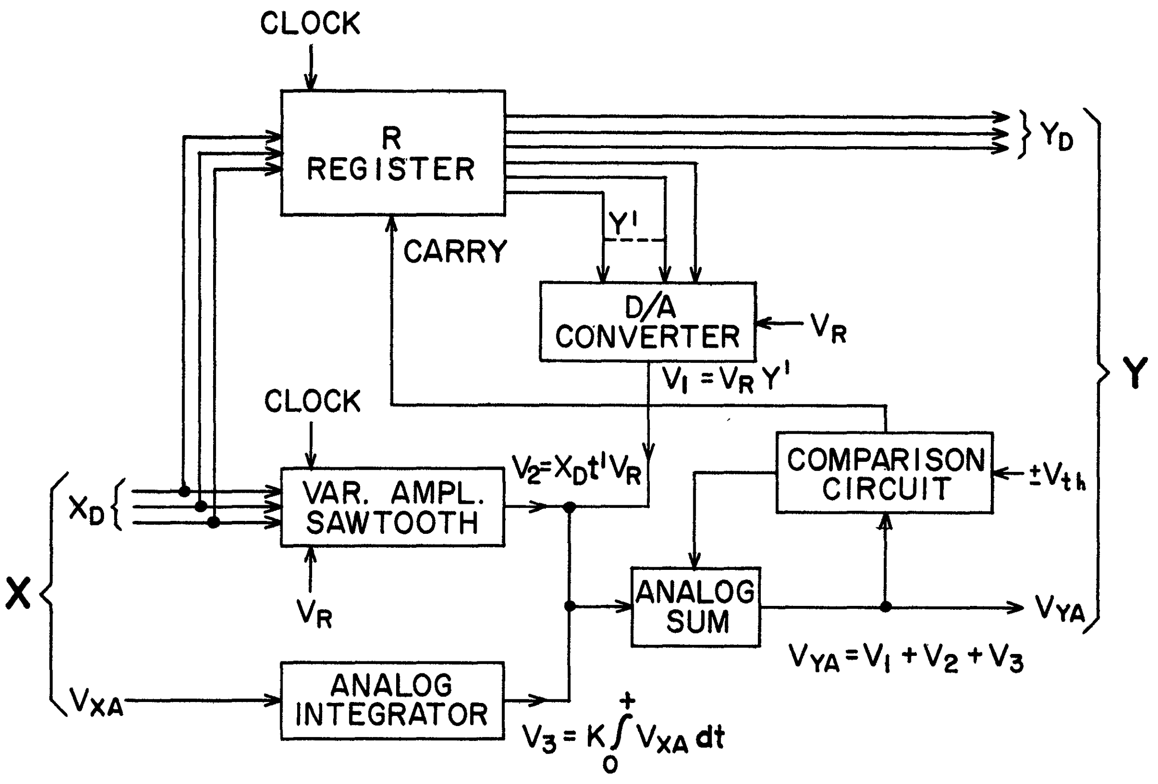


Fig. 8. Block diagram of the hybrid integrator.

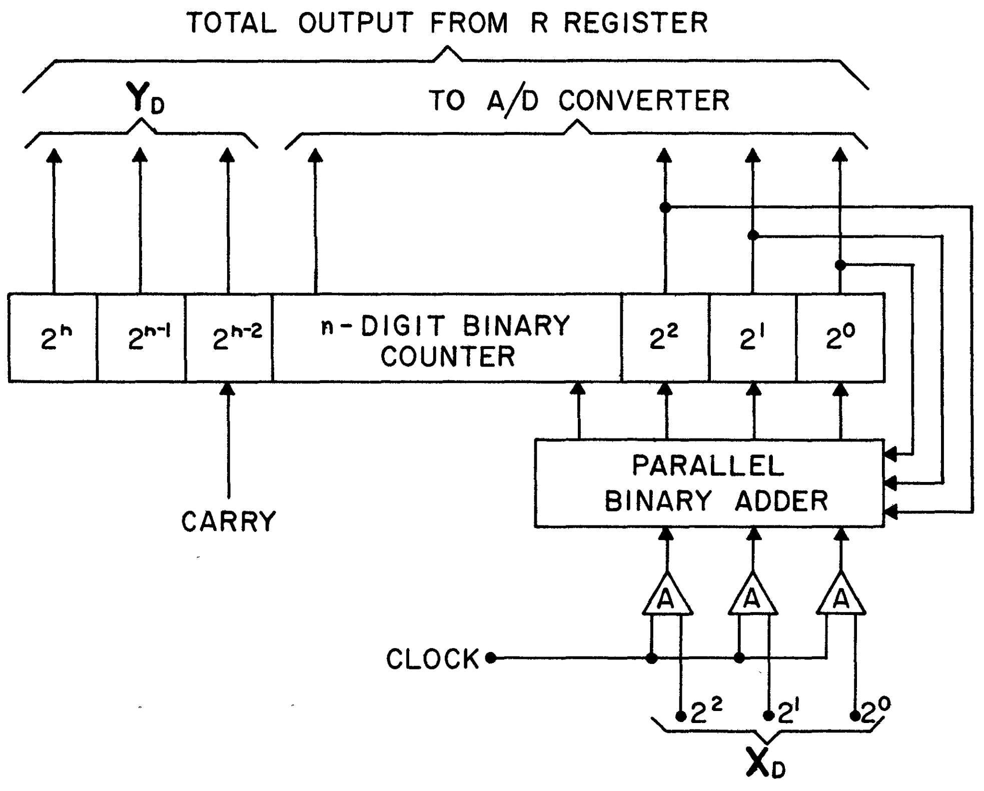


Fig. 9. Block diagram of the R-register.

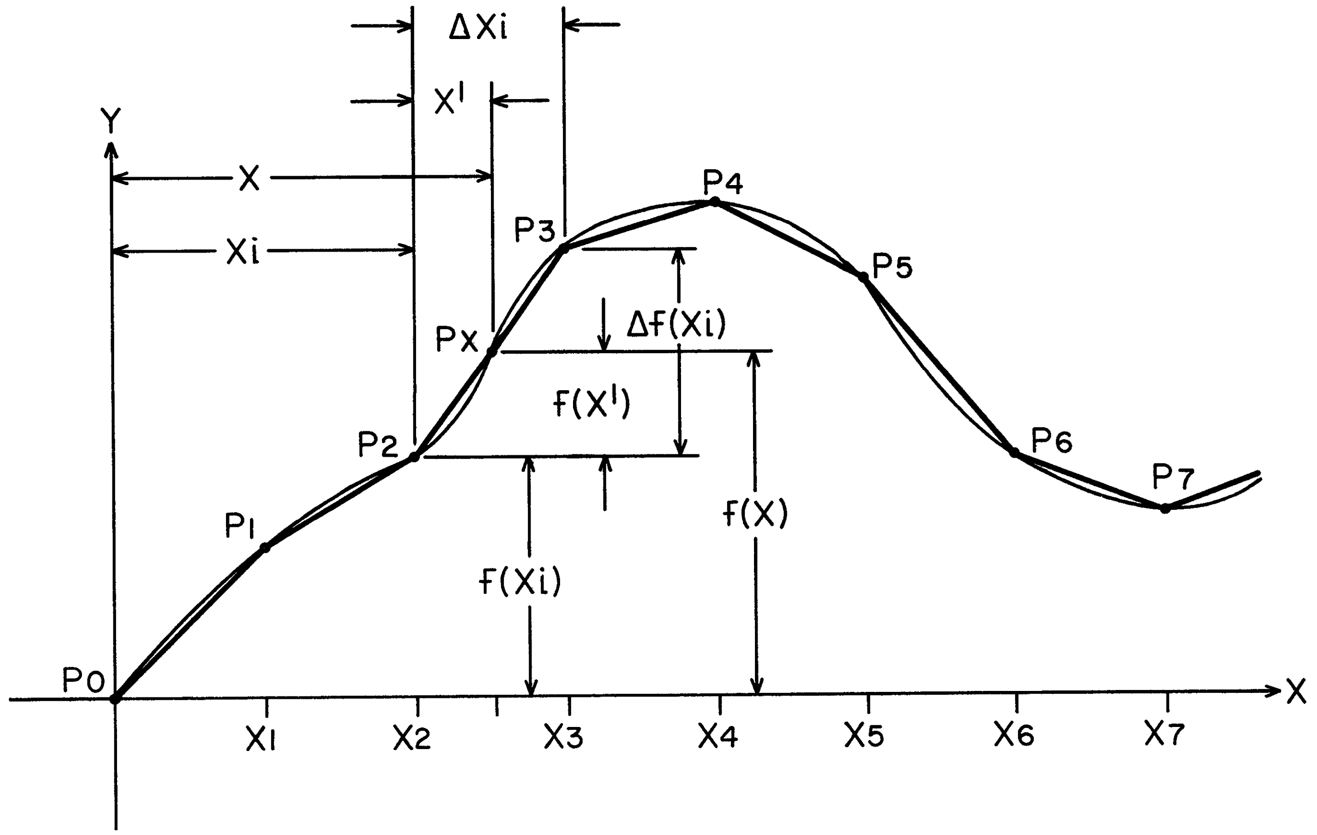


Fig. 10. Linear segments approximate the curve Y = f(x).

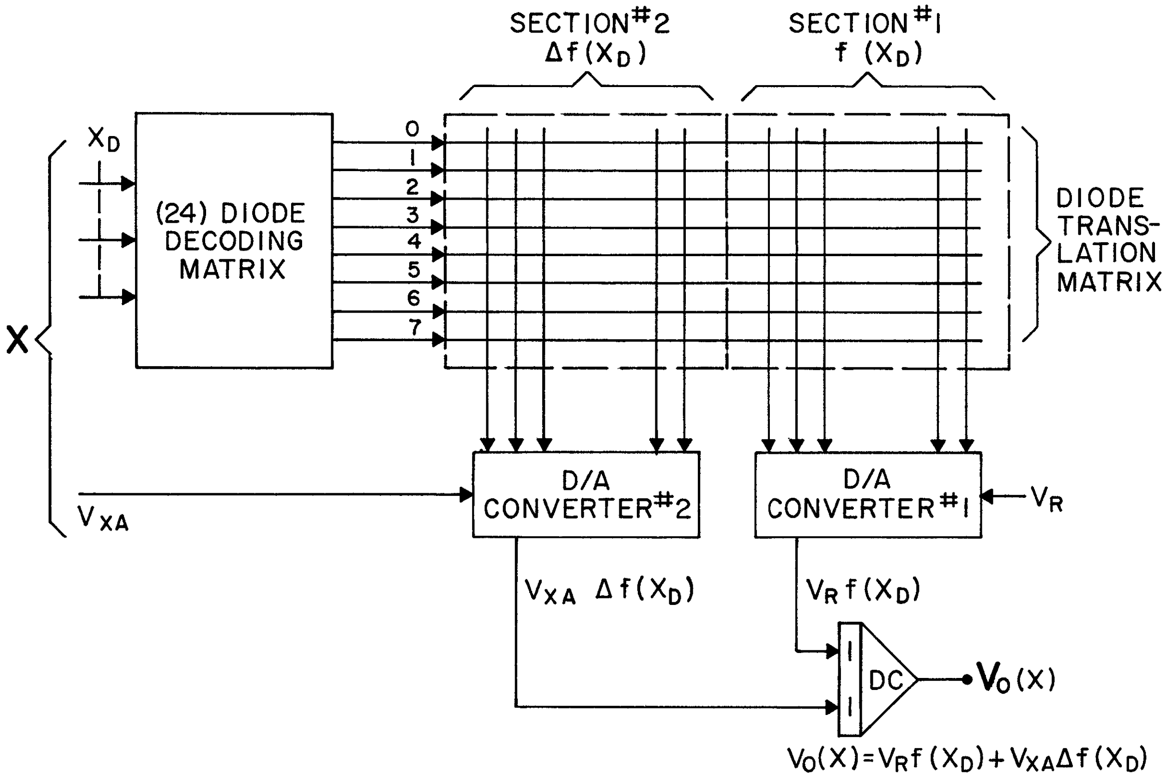


Fig. 11. Linear-segment hybrid function generator with hybrid input and analog output.

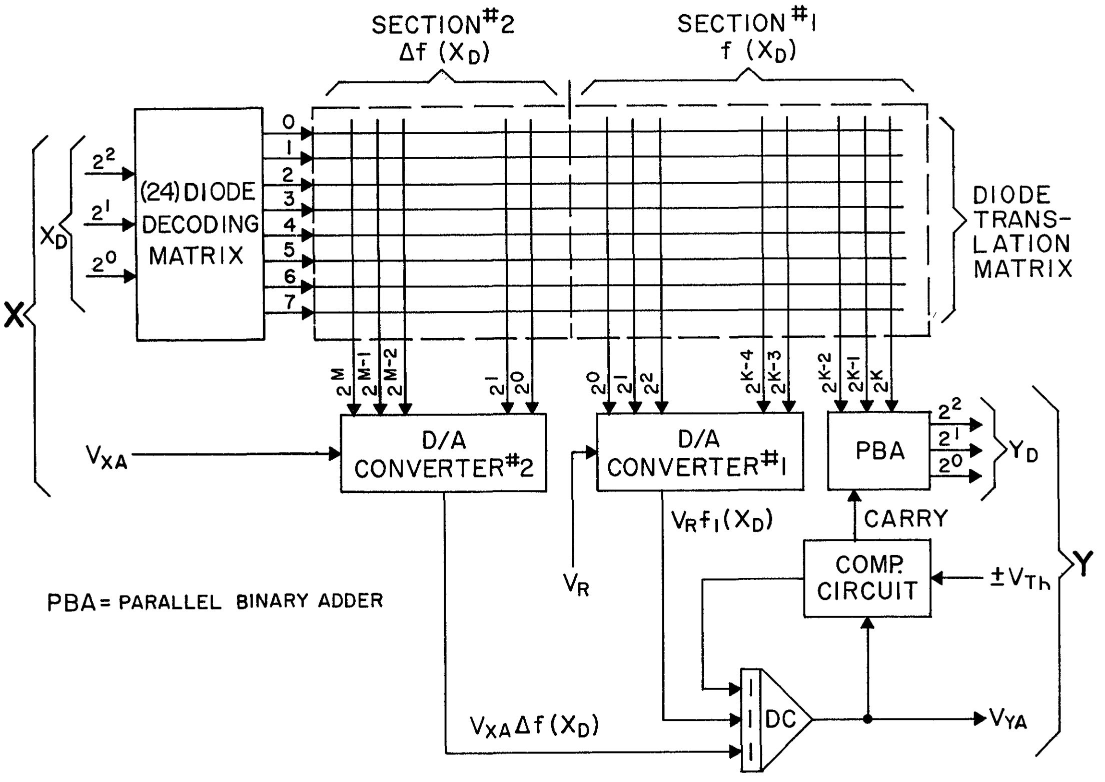


Fig. 12. Linear-segment hybrid function generator with hybrid input and output.

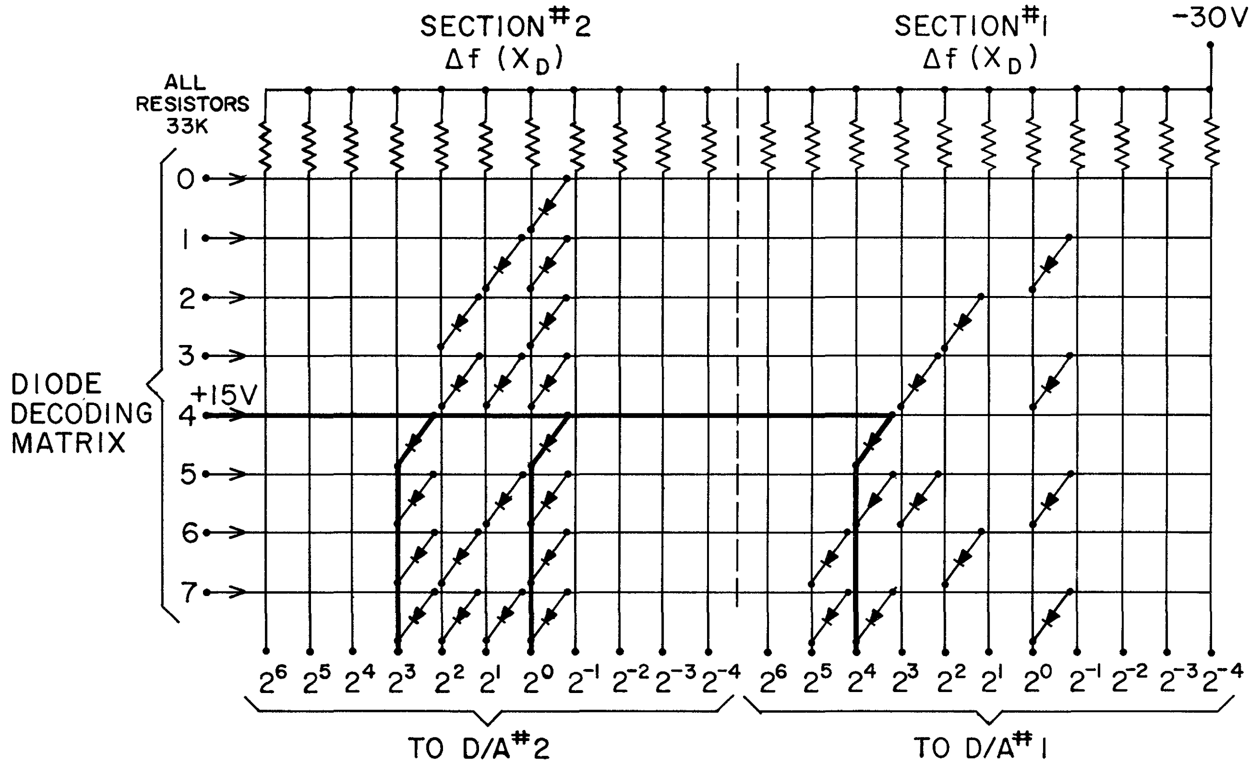


Fig. 13. Diode translation matrix wired for $f(x) = x^2$

OPTIMIZATION OF ANALOG COMPUTER LINEAR SYSTEM DYNAMIC CHARACTERISTICS

By Charles H. Single and Edward M. Billinghurst
BECKMAN/Berkeley Division
Richmond, California

Summary

The characteristics of the linear computing elements of the general-purpose electronic differential analyzer are discussed. Equivalent circuits are given for these elements. Criteria for optimization of the linear computing system in order to obtain maximum computational accuracy are given. Examples of the effects of optimization in improving system stability, transient response, and increasing bandwidth of high and medium accuracy computation are shown.

Introduction

The modern electronic differential analyzer (EDA) is widely used for solutions of various linear and non-linear mathematical equations and for the simulation of complex physical systems. The modular construction of such computers allows a large problem range.[1] Optimization of individual unit characteristics does not always lead to best overall computer performance; therefore, unit design must be undertaken with as complete a knowledge of final system configurations and accuracy requirements as possible.[2, 3, 4, 5]

This paper reviews proven design techniques that have been used to optimize the linear system dynamic accuracy of the EDA. The techniques have resulted in simultaneous improvement in these important system areas:
1) stability margin, 2) transient response (both small and large signal), 3) bandwidth of high accuracy computation, 4) bandwidth of medium accuracy computation.

For the EDA, improved stability margin considerably increases the flexibility of the computer. Difficult or unusual problems involving high gain loops, remote equipment operation, peculiar amplifier feedback and input networks, etc., can be patched with essentially no tendency for system oscillation.

The improved transient response and extended bandwidth of high accuracy computation have considerably improved the most basic performance criterion: Computational accuracy. This should be of general interest since many of these techniques can easily be applied to existing computers at small cost.

With the recent extension of the EDA to also serve as an iterative differential analyzer (IDA), the bandwidth of medium accuracy computation becomes more important. The numerical techniques used with the IDA often result in repetitive steps to final problem solution. This is coupled with a desire for higher speed in the repeated partial-solution phase to minimize total problem time. The accuracy limitations associated with the extension of computing frequencies beyond a few hundred cycles per second are discussed.

Basic Considerations

Static accuracy for both the EDA and the IDA is achieved through high-gain, low-drift dc amplifiers, precise and stable resistive networks, and stable, high-resolution coefficient potentiometers. Dynamic accuracy and stability margin are of at least equal importance. For optimization of these ac characteristics the composite system closed-loop transfer function should approach its ideal most closely. Contributing ac factors include integrator capacitor stability, equivalent circuit of network resistors and capacitors, potentiometer capacitive loading, amplifier-system transfer characteristics, and amplifier output impedance. The interaction of these factors is further complicated by practical considerations such as system wiring capacity. All of these factors should be included in the system performance analysis.

The improved system stability margin of the optimized-computer is of great value for accurate simulation of complex and/or high-gain loops of either differential or algebraic form. It is also advantageous with simulations requiring unusual feedback configurations or large capacitive loads. The better accuracy comes from the fact that no additional stabilizing capacitors are needed to avoid system oscillation. Such capacitors can cause large dynamic error. However, for many problems the stability margin typical of the non-optimized computer is still sufficient. In either case, the principal error limiting the bandwidth of high accuracy computation will be caused by net system phase error, since even slight phaseshift errors produce significant dynamic errors.

The large effect of small undesired phaseshift can readily be shown by examination of a steady state sinusoidal signal with only small phase error. With error defined as the difference between the ideal and actual signal:

$$\delta(j\omega) = E_i(j\omega) - E_a(j\omega) = E_i[\sin\omega t - \sin(\omega t - \phi)] \quad (1)$$

The per-unit error due to this small phaseshift is:

$$\delta / E_i(j\omega) = \sin\omega t - \sin(\omega t - \phi) \approx \phi \cos\omega t \quad (2)$$

The maximum amplitude of the per-unit error is therefore equal to the phaseshift in radians:

$$[\delta / E_i(j\omega)]_{max} \approx \phi \quad (3)$$

In any problem the net phase error, ϕ_o, is found by

$$\phi_o = \omega \sum_{j=1}^{m} \tau_j + 2 \sum_{k=1}^{n} \zeta_k \tau_k = \omega\tau_o \quad (4)$$

where τ_o is the equivalent first order time constant formed by the algebraic sum of the inevitable undesired high frequency poles and zeros introduced by the practical system. The lead or zero terms are taken as positive and the lag or pole terms taken as negative with ζ_k the damping factor of any quadratic terms.

To hold the maximum amplitude error due to undesired phaseshift below 0.2% at 100 cps (0.12°) in a particular problem closed-loop, or between two signal voltages inside a loop, the equivalent first order time constant, τ_o, of the loop or signal path must be smaller than 3.2 μ sec. If there were only one undesired pole, the breakpoint frequency, $(f = \frac{1}{2\pi\tau_o})$, would have to be higher than 50,000 cps for this 0.2% dynamic error at 100 cps. In practice more than one undesired pole will exist; this requires individual breakpoint frequencies to be much higher than 50,000 cps.

A conventional method of achieving lower phase error without the high bandwidth cited above is to allow individual elements to have a quadratic characteristic with a damping factor less than 0.5. This method considerably reduces the stability margin and tends to cause oscillation in problems involving complex or high gain loops. A better technique avoids the underdamped quadratic characteristic by careful placement of a closed-loop dipole with the zero separated just sufficiently from its associated pole to mask the low frequency phase errors caused by the undesired poles. The amplitude peaking caused by the dipole can be limited to less than a few tenths of one decibel. Application of the technique will be discussed in detail later.

The amplitude error from an undesired pole is the difference between the ideal and actual magnitude. For small phase error ($\phi \approx \omega\tau \ll 1$), the amplitude error is:

$$\epsilon_p = |M_i| - |M_a| = 1 - \frac{1}{\sqrt{1 + (\omega\tau)^2}}$$

$$1 - \left(1 - \frac{(\omega\tau)^2}{2}\right) \approx \frac{(\omega\tau)^2}{2} \quad (5)$$

Thus, the phase error from one pole, δ_p, is much larger than the amplitude error, ϵ_p:

$$\delta_p \approx \omega\tau \gg \frac{(\omega\tau)^2}{2} \approx \epsilon_p \quad (6)$$

Hence, amplitude error can be neglected with respect to phase error within the bandwidth of high accuracy computation.

Passive Component Characteristics

The passive components in the linear system are resistors, capacitors, and coefficient potentiometers. Their characteristics will be reviewed briefly, followed by a more detailed discussion of operational amplifier-system characteristics, and then the optimized linear system results will be shown.

Computing Resistors

Most computing resistors at present are wire-wound because they have better long term stability than other types. This situation may well change with continuing development of other resistor types, e.g., metal-film. Conventional specifications such as value, stability, temperature coefficient, power rating, packaging and ac characteristics are somewhat inadequate. Value may vary with self-heating, and with time due to changes in mechanical stress of the wire. Thermal potentials may exist from even small temperature gradients. Encapsulation can aggravate self-heating effects and may not be necessary within normal humidity limits. The resistors should be evaluated as used in their expected environments to establish such specifications.

The importance of relative or absolute value is primarily determined by the grouping of resistors allowed in the computer. If resistors are limited to definite groups, only the relative match of resistors within each group is important in establishing basic dc accuracy limits for the computer system. The more flexible arrangement, where resistors can be conveniently assigned to any amplifier, requires precise matching throughout the computer system.

The ac characteristics of wire-wound resistors for frequencies below 100 Kc can be approximated by the circuit shown in Figure 1. The driving-point impedance of this approximation is:

$$Z_r\,(s) = \frac{E_r}{I_r}\,(s) =$$

$$\frac{R_{dc}\left(\frac{L}{R_{dc}}\,s + 1\right)}{LC\,s^2 + R_{dc}\,C\,s + 1} =$$

$$\frac{1}{C}\;\frac{s + \frac{R_{dc}}{L}}{s^2 + \frac{R_{dc}}{2}\,s + \frac{1}{LC}} \qquad (7)$$

For resistors within the conventional range of 100K to 1 meg, the time constant, $\frac{L}{R_{dc}}$, can be held to less than 10^{-7} $\sec^{-1}$ (breakpoint beyond 1.5 megacycles) whereas the capacitor time constant R_{dc} C will typically be between 2×10^{-6} and 4×10^{-6} $\sec^{-1}$(breakpoint as low as 40 Kc). This allows simplification of the resistor transfer characteristic to:

$$Z_r(s) = \frac{R_{dc}}{(R_{dc}\,C\,s + 1)} = \frac{1}{C}\;\frac{1}{s + \frac{1}{R_{dc}C}} \qquad (8)$$

In the summing amplifier, the undesired poles of the resistors contribute no low-frequency phase error if Z_f and Z_i are perfectly matched. Mismatch in resistor capacity can cause either phase lead or lag as determined by Equation 4.

For integrator amplifiers, the resistor shunt capacity must be kept as low as possible since there is no convenient way to cancel its error effect. Four picofarads of capacity in parallel with 1 meg input resistors causes 0.25% maximum error at 100 cycles. With integrator feedback capacitors restricted to values greater than 0.01 μf, 100 cps computational frequencies can only be achieved with 100K input resistors. Four pf in parallel with a 100K input resistor causes only 0.025% maximum amplitude error.

These considerations make the presently available wire-wound resistor ac characteristics acceptable, i.e., resistor phase-errors essentially cancel in summing amplifiers and are small enough with the 100K integrator input resistors normally necessary to achieve high computational frequencies.

<u>Integrator Capacitors</u>

The precision capacitor used for integration is a much less ideal component than the precision resistor, i.e., the practical capacitor approaches its ideal characteristic $Z_c\,(s) = \frac{1}{s\,C}$ much less closely.[4, 6]

Capacitors generally have a larger temperature coefficient, are less stable in value, and have more complex electrical characteristics than resistors.[7]

Most Polystyrene capacitors have a temperature coefficient of -0.01%/°C to -0.012%/°C. To avoid value variation with ambient temperature changes they are generally mounted in an oven. The oven temperature usually is set above expected maximum ambient temperature to avoid the expense of a cooling system. Capacitor leakage increases with temperature, so the oven temperature should be set for the lowest temperature consistent with ambient requirements.

Even carefully selected capacitors in a constant environment show variation in value with time. Thus, it is desirable to be able to readjust or pad the capacitors without causing thermal disturbance.

Variations in temperature coefficient between capacitors and the fact that some capacitors show a temperature retrace error, i.e., they do not return to the same exact value when returned to precisely the original temperature, make the avoidance of temperature change desirable during adjustment. Another physical characteristic that affects value is the change in mechanical stresses with applied voltage.

The electrical characteristics are quite complex as shown by the equivalent circuit of Figure 2.[4,7] This 1 μf equivalent circuit is valid for $0.001 \text{ cps} \leqslant f \leqslant 100 \text{ cps}$. It is consistent with these easily measured capacitor error effects:

1) integrator drift increases with voltage level, due to dc leakage
2) effective leakage increases with frequency, due to dissipation factor
3) capacitor value reduces with frequency, due to dielectric characteristics
4) integrator output varies with capacitor history, due to dielectric absorption.

The simplified equivalent circuit, Figure 3, is useful for steady-state sinusoidal analyses. The leakage resistance, R_L, varies inversely with frequency. The capacity value can be considered constant for many analyses, as the variation is only about 0.02% per decade frequency change. However, this variation in value with frequency is quite important in adjusting or padding the capacitor to its "precise value." For computer use, capacitor value should be determined at frequencies consistent with those used in the computer. This can conveniently be done with the various resistance-time measurements where the time constant of the capacitor resistor combination is determined.[6,7,8] Capacitor value is then based on the same resistance standard used for computer resistor measurement. Details of these electrical characteristics are given in Appendix 1.

Coefficient Potentiometers

For extension of the bandwidth of high accuracy computation, the phaseshift from potentiometer input to wiper arm is an important consideration.[9] An equivalent circuit for the typical computer multi-turn, copper-mandrel, coefficient potentiometer is shown in Figure 4.

C_o of Figure 4 is essentially constant for all potentiometer settings, and only slightly dependent on potentiometer value. This equivalent circuit is valid within the limits of Table I.

Table I

R	C_o	Equiv. Ckt. * Freq. Limit (cps)	Figure 5 Freq. Limit (cps)
10K	200pf	$1.1(10)^3$	$6.6(10)^3$
20K	204pf	$550(10)^3$	$3.3(10)^3$
30K	208pf	$350(10)^3$	$2.1(10)^3$
50K	216pf	$200(10)^3$	$1.2(10)^3$
100K	224pf	$82(10)^3$	590

* The equivalent circuit frequency limit is derived from potentiometer experimental step function data.

The transfer function of the equivalent circuit is

$$\frac{E_o}{E_i}(s) = \frac{\gamma[(1-\gamma) R C_o s + 1]}{\gamma(1-\gamma) R (2 C_o + C_{ext}) s + 1} = \frac{\gamma(\tau_1 s + 1)}{\tau_2 s + 1} \tag{9}$$

where γ is the pot arm displacement from the grounded end.

Equation 9 is a simple lead-lag, or lag-lead transfer characteristic. Phaseshift, ϕ, for sinusoidal signals is $\phi = \tan^{-1}\omega\tau_1 - \tan^{-1}\omega\tau_2$. (10)

However, for both EDA and IDA error analysis the frequencies involved allow further simplification. For $\omega\tau_1 < 0.1$ and $\omega\tau_2 < 0.1$, ϕ can be approximated by $\omega(\tau_1 - \tau_2)$:

$$\phi \approx \omega\tau_{net} = \omega(\tau_1 - \tau_2) = \omega R C_o (1-\gamma)(1-2\gamma-k\gamma) \tag{11}$$

where $k = \dfrac{C_{ext}}{C_o}$.

Universal curves showing normalized phaseshift $\phi / \omega R C_o$, are given in Figure 5. These curves are valid for any potentiometer value, displacement, capacitive load, and frequency within the limits of Table I.

The amplitude error at low frequencies

can be obtained from Equation 9 as:

$$\epsilon = \gamma - \gamma \left[\frac{1 + (\omega\tau_1)^2}{1 + (\omega\tau_2)^2}\right]^{1/2} \tag{12}$$

$$1/2\ \gamma\ (\tau_2^{\,2} - \tau_1^{\,2})\ \omega^2 \tag{13}$$

Comparison of Equations 11 and 13 shows that amplitude error is smaller than phase error by an order of magnitude in the frequency variable, ω. Thus the potentiometer error is adequately represented by considering only phase error if $\omega\tau_1 < 0.1$ & $\omega\tau_2 < 0.1$.

From the simple equivalent circuit of Figure 4 it would be easy to compute a compensating capacitor placed from potentiometer wiper to either potentiometer input or ground that would exactly cancel phase error for any particular displacement and resistive-capacitive loading. However, this is not practical with the large number of coefficient potentiometers of various displacements and loadings.

A more practical solution to this problem is to compensate the potentiometers by using tap-capacitors to avoid displacement dependence.[10, 11] This makes it possible to achieve excellent square wave response to fast-rise step inputs and much improved phase error. Details of these techniques are given in Appendix 2. A few such capacitively compensated potentiometers will prove invaluable for critical simulations in even a moderate-sized computer.

From the above discussion it is apparent that higher potentiometer computational accuracy can be achieved by minimizing both external load capacity and potentiometer value.

At dc the potentiometer transfer characteristic is a function of displacement, γ, and resistive load, R_L:

$$\frac{E_o}{E_i} = \frac{\gamma}{1 + \gamma(1 - \gamma)\alpha} \tag{14}$$

where $\alpha = R_P / R_L$.

The slope as a function of displacement is:

$$\frac{\partial E_o\ E_i}{\partial \gamma} = \frac{1 + \alpha\gamma^2}{[1 + \gamma(1 - \gamma)\alpha]^2} \tag{15}$$

At $\gamma = 1$ the largest slope is found:

$$\left.\frac{\partial E_o\ E_i}{\partial \gamma}\right|_{max} = \alpha + 1 \tag{16}$$

If coefficient-setting accuracy of one part in 10,000 is desired and minimum R_L is 100K:

Rp	α max.	max. slope	÷ Set. Acc'y. / 2	= * Needed No. of Turns	Std. No. of Turns (±10%)	Std. req. (%)
30K	0.3	1.3	$\frac{1}{5,000}$	6,500	11,119	172
50K	0.5	1.5	$\frac{1}{5,000}$	7,500	11,950	159

* Maximum setting error is equal to 1/2 potentiometer resolution.

If standard potentiometers are used, the 30K value is preferred since it has 40% less computational error with the same capacitive load. Both potentiometers have adequate resolution.

Operational Amplifier System Characteristics

The operational amplifiers in an analog computer have important effects on the dynamic accuracy of linear computation. A knowledge of the performance of operational amplifiers and their associated passive networks in a large computer is necessary in order to make judicious choices in amplifier and system design for optimum results. The definition of optimum may vary with the particular problems to be solved. It may mean high accuracy at slow computing speeds, or medium accuracy at faster computing speeds. In any event, accuracy will tend to decrease as higher frequencies are encountered in the problem solution.

General Purpose Computer Amplifiers

Computer amplifiers are primarily used to perform the operations of summation, integration with respect to time, and multiplication by a constant. In addition, they may be used with special networks as limiters, transfer function generators, dividers or square root extractors (in association with computing multipliers), inverse function generation, etc. The amplifiers are always operated in a feedback or closed-loop configuration. As a result, their open-loop frequency response characteristics must be designed so that the amplifiers are stable under any permissible computing configurations. They must also have low dc drift and low grid-current. [1] As operational amplifiers, they must accurately perform their mathematical functions. Accuracy of computation requires high open-loop gain; stability requires that the gain be reduced in a fairly restricted manner. Thus, a compromise must be made between accuracy and stability.

Open-Loop Transfer Function

Figure 6 is a block diagram of a typical dc amplifier with chopper stabilization to minimize drift effects. The open-loop transfer function is given by:

$$\frac{E_o(s)}{e\ (s)} = G\ (s) = (G_1\ (s) + G_2\ (s)\ )\ G_3\ (s) \tag{17}$$

G_1 and G_3 can be calculated from standard linear vacuum tube and network theory. G_2 can be satisfactorily approximated by considering the chopper modulated ac section as a frequency-independent amplifier associated with the input and output filters. $G_1 + G_3$ contribute a cluster of equal numbers of poles and zeros fairly close to the origin. G_3 will usually have a pole in the neighborhood of 10 to 100 cps and other poles in the range between 100 Kc to several Mc. These are in addition to those contained in $(G_1 + G_2)$. The low-frequency pole is inserted to obtain a frequency response approximating a 20 db decade roll-off until the gain is below zero db. If the low frequency pole is at 10 cps with gain at that frequency of 10^5, the frequency response goes through unity gain at one Mc.

Once G_1, G_2 and G_3 are known, the amplifier open-loop pole-zero plot can be drawn. A typical pole-zero configuration is shown in Figure 7. The poles and zeros nearest the origin are due to $G_1 + G_2$, the fourth pole is the low frequency pole of G_3. The dotted dipole combinations represent incomplete compensation of inter-stage coupling networks. These generally have negligible effect on the closed-loop response providing they are not widely separated. The cluster of poles far to the left represents the high-frequency poles of G_1 and G_3. Since several decades separate the lowest and highest poles, the drawing is not to scale, but merely indicates the relative positions.

Determination of Closed-Loop Transfer Function

The amplifier, as actually installed in the computer, is represented by the diagram of Figure 8a. Figure 8b is the current-generator equivalent circuit.

Z_i is the input impedance from a particular signal source, E_i.

Z_g is the impedance from summing junction to ground.

Z_f is the feedback impedance.

Z_o is the output impedance of the final stage of the amplifier.

Z_L is the load impedance, including the output to ground capacity of wires connected to the output terminal. $-G(s)$ is the open-loop, open circuit gain of the amplifier.

The following equations can be written from Figure 8b:

$$-Ge\ Y_o = E_o\ (Y_o + Y_L + Y_f) - eY_f$$

$$E_iY_i = -E_o\ Yf + e(Y_i + Y_g + Y_f)$$

Where $Y_k = 1/Z_k$

Substitute $Y_a = Y_i + Y_g + Y_f$

$$Y_b = Y_o + Y_L + Y_f$$

Solving for E_o/E_i, we obtain:

$$\frac{E_o}{E_i} = \frac{Z_f}{Z_i}\left[\frac{\dfrac{GY_f\ Y_o}{Y_a\ Y_b - Y_f^2}}{1+\dfrac{GY_f\ Y_o}{Y_a\ Y_b - Y_f^2}}\right]\left[1 - \frac{Y_f}{GY_o}\right] \tag{18}$$

The extension to multiple inputs results in the following expression:

$$E_o = \left(- \sum_{i=1}^{n} \frac{E_i Z_f}{Z_i}\right) \left[\frac{\dfrac{GY_f Y_o}{Y_a Y_b - Y_f^2}}{1 + \dfrac{GY_f Y_o}{Y_a Y_b - Y_f^2}} \right] \left[1 - \frac{Y_f}{GY_o} \right] \qquad (19)$$

Where $Y_a = Y_g + Y_f + \sum_{i=1}^{n} Y_i$.

The first bracket of Equation 18 is the ideal mathematical operation to be performed. The second bracket determines stability and error in computation. For most computing configurations, the third bracket $\left[1 - \frac{Y_f}{GY_o}\right] \approx 1$, so this term can usually be ignored. It is necessary to consider it if the response to very high frequencies (greater than approximately 300 Kc) is of interest. We note also that $1 - \frac{Y_f}{GY_o}$ will not generally produce a right-half plane pole, so it does not affect the stability.

Z_i is usually a resistor with characteristics as discussed earlier. Z_f will be the same type of resistor as Z_i, or an integrating capacitor. Z_g is the input impedance to the amplifier, including the chopper section, in parallel with the capacity of wiring connected to the grid or summing junction. The amplifiers in a large computer must be located some distance away from the patchboard and passive elements. As a result, the summing junction capacity, C_g, is usually of the order of 500 to 1000 pf.

Z_o will be adequately represented by a resistance if the output stage is designed for low impedance. This is generally a requirement in a computing amplifier to allow it to deliver several milliamps output current.

Y_a is the parallel admittance of Y_i, Y_g and Y_f. It can usually be replaced by a parallel R-C circuit. Y_b can frequently be represented by a single R-C circuit since it is the parallel admittance of Y_f, Y_o and Y_L. However, Y_b may be more complicated than a single parallel R-C circuit if Y_L is not a pure R-C combination. For typical summer configurations, C_g will be large compared to C_i and C_f, so these quantities will usually have small effect on Y_a. Similarly C_L dominates C_f in Y_b.

Returning to Equation 18, the expression in the first bracket, is the transfer function desired for the solution of problems. This expression is modified by the second bracket, thus introducing extraneous roots in the problem solution. For accurate computation, the extraneous roots must be kept as far out as possible.

The middle bracket expression of Equation 18 is of importance in determining the amplifier closed-loop extraneous poles and zeros. It also indicates the amplifier stability. This expression is well suited to analysis by root-locus techniques,[13] especially for qualitative determination of the effects of external summing junction and load capacity.

Figure 9 is a root-locus plot for unity gain summer using 1 M resistors. This is a typical configuration found in a general-purpose computer, where the summing junction and load capacities are fairly large due to the long lengths of interconnecting wires required. We see here that the dominant portion of the root-locus is bowed somewhat due to the action of the zero produced by Y_f. Y_b is a pole primarily due to the load capacity and open-loop output impedance. Y_a is a pole due to the input and feedback resistors in parallel, shunted by the grid to ground capacity. Obviously, as either C_L or C_g increases, the root-locus will be displaced more to the right, and the closed-loop poles will show less damping. This root-locus demonstrates the fact that the external networks and wiring capacity have a major effect in determining the actual summer response in the system.

The underdamped characteristic of the configuration of Figure 9 is undesirable due to low bandwidth and excessive phaseshift. Amplifiers of this type are particularly unsatisfactory when used in problems involving multiple closed-loops, e.g. in high gain algebraic loops. Multiple loop configurations have a large tendency toward self-oscillations.

Figure 10 shows a root-locus plot for a summer which has extra capacitors paralleled across Z_f and Z_i. Proper selection of these capacitors gives an over-damped response with much wider bandwidth: e.g. 30 - 50 Kc instead of 12 - 15 Kc, and no peaking. The compensation also produces a dipole in the vicinity of the zero caused by Y_f. This dipole adds a slight phase lead to help reduce low frequency phaseshift. Multiple loops using these compensated amplifiers have much less tendency to oscillate. The amplifiers can also drive larger capacitive loads then the uncompensated type. This is important when amplifiers must drive remote equipment. Load capacity can be increased by at least a factor of 10. Capacitor values must be selected so that Z_f and Z_i have equal time constants. That is, X10 (100K) resistors are compensated with 10 times the capacity associated with X1 (1M) resistors. Addition of compensating capacitors is very important in increasing overall linear system stability and the bandwidth for a given accuracy of computation. It should be noted that input resistors associated with integrators must have no compensating capacitors, since in this case the capacitors will cause additional undesirable phase error.

The selection of the proper values of compensating capacitors for the computing resistors is based on optimizing overall-performance of coefficient potentiometer-amplifier response, rather than amplifier response alone. Summers are very often driven from coefficient pots, and the capacitor associated with the amplifier input resistor loads the pot. Therefore, the compensating capacitor value is chosen to give minimum phase error and maximum bandwidth from pot input to amplifier output. In computers using 30K pots, the best values of compensation are about 43 pf with 1M resistors, and 430 pf with 100K resistors.

Conclusions

We have discussed the linear computing elements from the standpoint of their actual equivalent circuits and their relationships to computing accuracy, system stability, and system bandwidth.

Computing resistors are shown to require minimum shunt capacity in their construction, and the shunt capacity must be equalized for all resistors. Minimization of the capacity is required for small phase errors in integrators. Capacity match is required for small phase errors in summer amplifiers. They should have a means of adjustment if their value is likely to change with age.

Computing capacitors must be stable in value, have large leakage resistance, and minimum dissipation factor and dielectric absorption (soakage) effects. They should be installed in a controlled temperature environment and be easily adjustable.

Coefficient potentiometers require high resolution for accurate settings, and low resistance value to minimize capacitive loading errors.

Operational amplifiers must be designed to be stable under any permissible computing configuration in the computer system. In addition, summing resistors should be capacitively compensated to produce maximum bandwidth and stability in multiple loops and when the amplifier is driving large capacitive loads.

The following examples indicate the improved computer performance obtained by the careful consideration of the design of the linear system. The data was obtained from a standard BECKMAN EASE® 1100 Series computer using compensated resistors, 30K coefficient pots, Polystyrene computing capacitors and EASE® Model 1148 Operational Amplifiers.

Improved summer amplifier bandwidth is shown by Figure 11. The absence of overshoot indicates an overdamped (dominant first-order) system. The rise time of about 15 μ sec indicates a bandwidth of approximately 30 Kc.

Figure 12 shows the ability of a summer to operate with large summing junction or load capacity. The step response is still well damped with as much as 0.3μfd load capacity. The uncompensated amplifier would oscillate with as little as 0.02μfd.

Figure 13 shows the stability of loops with up to 19 unity gain amplifiers. Unity gain loops will oscillate with as few as 3 to 5 uncompensated amplifiers.

Figure 14 illustrates the gain possible in a three-summer loop with a coefficient pot. The response is still damped at a loop gain of 8, while uncompensated amplifiers will oscillate at a gain of less than 2.

The steady-state error of a three-amplifier oscillator is easy to measure. It uses all of the basic components of the linear system: Resistors, capacitors, coefficient potentiometers, and amplifiers. Therefore, it is useful in demonstrating the balanced or optimized system design.

The three-amplifier oscillator uses two integrators, one summing amplifier, and one to two coefficient potentiometers. The frequency of the oscillator varies with loop gain, $f = \sqrt{\text{loop gain}}/2\pi$.

The steady state error occurs as a slight divergence or convergence in voltage amplitude. This can be measured over many cycles, or by reading successive peak voltages. Both techniques are useful over the frequency range of interest $0.001 \leqslant f \leqslant 100$ cps. The data can be plotted more conveniently as damping, $\zeta = \alpha / \omega_n$, i.e., amplitude decrement per radian.

Figure 15 shows component damping, vs. frequency for a three-amplifier oscillator with a maximum loop gain of 10^5 which gives a maximum frequency of 50.4 cps. Note that the loop gain, or frequency, is varied by coefficient potentiometer displacement, γ. The capacitor loss is approximated by a constant dissipation factor of 1.5×10^{-4}, causing a frequency independent convergent effect. The resistor ac mismatch can cause either a convergent of divergent effect as determined by the net lead or lag in summing amplifier. Note that phase error effects increase with frequency. The large spread is for 3μ sec net mismatch; the smaller spread is for 2μ sec mismatch. Current production achieves less than 2μ sec worst case mismatch, or less than ± 1.0 pf spread in the one meg capacitively compensated resistor. A slight lead or convergent effect comes from the two integrator input resistors. The potentiometer effect is initially convergent due to lead then divergent due to lag, and finally zero at full displacement, $\gamma = 1$. The potentiometer data is for uncompensated 30K potentiometers under 660 pf capacitive load. This error effect could be reduced by a factor of 5 with the two-tap compensated potentiometer, i.e. essentially eliminated.

Figure 16 shows the combined errors of Figure 15. Note that phase error of the uncompensated potentiometer causes the oscillator to go divergent between 22 and 34 cps. Typical cross-over frequency for non-optimized computers is between 4 and 5 cps. This is very easy to check, since the oscillator amplitude is neither convergent nor divergent at the cross-over frequency.

A loop gain of 10^4 yields an f_{max} of 16 cps which is more common for the EDA. The potentiometer error is too large at loop gains of 10^5 as shown above. For such high gain loops compensated potentiometers should be used. $(10)^5$ loop gain will occur more often in IDA problems. Figures 16 and 17 show the balanced error that is achieved for the EDA at loop gains less than 10^4. Note that the capacitor dissipation effect offsets the damping curve approximately the same amount that the summing amplifier resistor mismatch spreads the curve, and approximately the same amount the potentiometer capacitive load distorts the curve. It is on this basis that optimum design is claimed. No error source is over-emphasized, and the excellent system stability margin has been achieved.

Although there is a rather large class of problems for IDA application that do not depend on dynamic accuracy for correct solution, there is an even larger class of problems whose accuracy will be severely limited if dynamic accuracy is poor. Caution should be exercised in extending IDA computational frequencies to obtain faster problem solution, unless careful attention is paid to the dynamic error factors reviewed in this paper.

APPENDICES

1. Capacitor Characteristics

Dc Leakage

The dc leakage of the integrator capacitor limits integrator accuracy at low frequency and causes drift of non-zero integrator voltages (in the HOLD mode). Both effects can be used as a convenient means of capacitor leakage measurement in the computer:

$$R_{dc} \approx \frac{1}{\alpha C} \qquad \text{(A. 1. 0)}$$

where α is the exponential decay coefficient for a three-amplifier sine-generator. The frequency should be below 0.001 cps and the circuit should use equal capacitors in the two integrators. See Figure A. 1. 0.

In the HOLD mode

$$R_{dc} \approx \frac{E_{max}}{C} \quad \frac{1}{(dE/dt)_{max}} \qquad \text{(A. 1. 1)}$$

E_{max} should be both plus and minus to cancel the effects of amplifier input grid current and grid to ground current due to offset. Averaging the two initial decay rates will yield a valid R_{dc} only if the E_{max} is established (in I. C. mode) for a few minutes to avoid dielectric absorption effects.[7] R_{dc} is typically 10^{12} to 10^{13} ohms for Polystyrent $1 \mu f$ capacitors.

Dissipation Factor

Dissipation factor, D, is conventionally used to evaluate dielectric materials for capacitors. βIt is a steady-state sinusoidal voltage characteristic defined as

$$D = \frac{\text{Energy loss radian}}{\text{Peak energy stored}} \qquad \text{(A. 1. 2)}$$

$$\frac{1}{\omega R_L(f) C(f)}$$

where C (f) and R_L (f) are frequency dependent, and paralleled to form the steady-state equivalent circuit of the capacitor, Figure 3. A three-amplifier sine-generator with only capacitor loss (no net phase error) yields a per radian amplitude decrement, $\zeta = \alpha/\omega$, equal to the capacitor dissipation factor, D.

For Polystyrene capacitors C (f) varies only about 0.02% per decade frequency change and R_L varies inversely with frequency. For $1 \mu fd$ Polystyrene capacitors

$$R_L \approx (10)^{-9}/f \qquad \text{(A. 1. 3)}$$

between $0.01 < f < 100$ cps, or $D \approx 1.5\,(10)^{-4}$. Figures 15 through 18 in the main report use the steady-state approximation of capacitors in computing capacitor error.

Figure A. 1. 1 shows dissipation factors for various materials at a lower frequency range than typically specified, i. e. computer frequencies.

Figure A. 1. 2 shows dissipation factor data for a typical Polystyrene $1 \mu fd$ capacitor. A selected Teflon capacitor's data is also shown to indicate possible reduction of this important dynamic error.

Figure A. 1. 3 shows the normalized dissipation factor of a large, perfect capacitor shunted by a series resistor and capacitor. It is possible to approximate the practical capacitor with a series of such resistor-capacitor shunts to achieve an approximately constant dissipation factor, as shown.

Equivalent Circuit

Figure A. 1. 4 shows the application of the technique of Figure A. 1. 3 to the measured Polystyrene dissipation factor of Figure A. 1. 2. This is the basis of the fixed-parameter equivalent circuit of Figure 2 in the main report. The fixed-parameter circuit is somewhat complex, but can be used for transient studies.

Value Variation with Frequency

Figure A. 1. 5 shows the capacity change predicted from the equivalent circuit as the solid line. Note that the experimental points confirm the equivalent circuit's validity from this viewpoint. Approximately 0.02% variation in value per decade frequency change is observed.

Dielectric Absorption

Limited transient evaluation of the equivalent circuit has been done to explain dielectric absorption or history effects. The analysis is somewhat difficult, as the equivalent circuit is complex. It is sufficient for this paper to point out that it is easy to get 0.02% to 0.04% differences in answers using $1 \mu fd$ capacitors even at low computational frequencies ($f < 0.1$ cps) due to capacitor history. A typical example is this: Soak the capacitor at +100 V in a false I. C. mode; establish zero volts I.C., for 10 to 30 discharge time constants, and then integrate a small step input in the COMPUTE mode. The "ramp" integrator output voltage will differ 20 to 40 mv if the false I.C. is changed to -100 volts. After the first 15 to 20 seconds the difference is independent of time. This false I.C. can easily be a voltage computed in a previous cycle and soaked in a long HOLD interval before the next compute cycle is started.

Correspondingly, it is very difficult to HOLD a voltage after a rapid transient is computed. The long time constant terms of the equivalent circuit of Figure 2 (in the main report) will remain charged to essentially the I.C. voltage during the compute interval. They will later reduce the voltage 0.02% to 0.04% as the HOLD interval continues. The basic answer to this problem is a better dielectric material than Polystyrene for the integrator capacitors. Teflon capacitors offer promise of some improvement, but are not yet in general use.

2. Copper-Mandrel Potentiometer Characteristics and Compensation [9]

Figure A.2.0 shows the circuit used to determine the equivalent circuit of the coefficient potentiometer. The oscilloscope was used only as a phase null device, with null achieved by addition of C_x or C_y for the particular potentiometer displacement.

The Type A Helipot® potentiometers checked (10K, 20K, 30K, 50K and 100K) were symmetrical, i.e. C_o equaled C_1, which allows this simple computation:

$$C_o = C_1 = \frac{\gamma (C_s + C_x) - (1 - \gamma) C_y}{1 - 2\gamma} \qquad \text{(A.2.0)}$$

where γ is pot displacement, C_s is oscilloscope input capacitance, and either C_x or C_y is zero at any given displacement. For other manufacturer's potentiometers, construction details may cause a slight difference between C_o and C_1. This difference is not important since C_o and C_1 values can be similarly computed. The curves of this report can be used by combining any difference between C_o and C_1 with the load capacity, C_e.

C_o turns out to be invariant for all values of displacement, $0.05 < \gamma < 0.95$. This results in the very simple equivalent circuit of Figure 4 and the normalized phase error vs. displacement of Figure 5.

Potentiometer capacitive loads generally are limited to $1 \leqslant k = C_e \; C_o \leqslant 3$ in the computer. From Figure 5, the normalized phaseshift, $\phi/\omega R C_o$, varies between +1.0 and -0.33 for k = 1, between +1.0 and -0.8 for k = 3. The resulting phase error at 100 cps for 30K and 50K potentiometers is:

Rp	$k=C_e/C_o$	$\phi/\omega RC_o$	ϕ (Radians)	ϕ (Degrees)
30K	1.0	+1.0	+0.0039	+0.22
30K	1.0	-0.33	-0.0013	-0.075
30K	3.0	+1.0	+0.0039	+0.22
30K	3.0	-0.8	-0.0031	-0.18
50K	1.0	+1	+0.0068	+0.39
50K	1.0	-0.33	-0.0023	-0.13
50K	3.0	+1	+0.0068	+0.39
50K	3.0	-0.8	-0.0054	-0.31

The largest positive phase error occurs at zero displacement. This particular displacement is of little interest in practice, but is approached with small potentiometer coefficients. Similarly, the largest negative phase error is restricted to displacements of $0.6 < \gamma < 0.7$. Therefore, "typical" potentiometer dynamic errors will depend on "typical" displacement values.

The amplitude error due to phaseshift is approximately equal to the phase error in radians, which means that the 30K potentiometer can cause up to 0.4% dynamic error at 100 cps. The capacitive load seldom will reach the $k = C_e \; C_o = 3$ value suggested here as an upper limit. If the typical value of k = 1 is used, and small pot displacements avoided, the "typical" dynamic error of 30K potentiometers is 0.1% to 0.15% at 100 cps: 50K potentiometers,is 0.17% to 0.25% at 100 cps.

In critical simulations it may be essential to eliminate potentiometer dynamic error. Various means of compensating for potentiometer phase error have been suggested. Most are suitable for canceling only potentiometer error, but not capacitively loaded potentiometer error.[10] One technique, that of placing small capacitors from a few distributed taps to either pot input or ground, readily allows extension to include arbitrary capacitive loads.[11] A few examples of this technique are given to show possible potentiometer phase error reduction. The analysis is not included here.[9] These curves should be compared with Figure 5.

Figure A.2.1 - Normalized phase error, $\phi/\omega R C_o$ vs. displacement γ; one tap at $\gamma = 0.5$. Tap-capacitor value is

equal to the load capacitor.

Figure A.2.2 - $\phi/\omega R C_o$ vs. γ; two taps at $\gamma = 1/3$ and $\gamma = 2/3$.

Figure A.2.3 - $\phi/\omega R C_o$ vs. γ; three taps at $\gamma = 1/4$, $\gamma = 1/2$, and $\gamma = 3/4$.

Figure A.2.4 - Shows an experimentally determined circuit for a ten-tap capacitively compensated pot with 940 pf capacitive load.

Figure a.2.5 - Shows the square wave response of the ten-tap pot compared with an uncompensated pot.

Figure A.2.6 - Shows the measured $\phi/\omega R C_o$ of the ten-tap pot compared with the uncompensated pot, both potentiometers with 940 pf capacitive load.

Figure A.2.7 - $\phi/\omega R C_o$ vs. γ; two taps at $\gamma = 0.1$ and $\gamma = 0.7$. Note that the non-uniform tap spacing results in less error than the uniform tap spacing of Figure A.2.2. For $0.05<\gamma<1.0$ and $k = 4.5$, the maximum $\phi/\omega R C_o$ is 0.24. This compares very favorably with the 0.18 maximum value of the ten-tap pot as shown in Figure A.2.6.

Restriction of $\phi/\omega R C_o$ to a maximum value of 0.2 for $k = 3$ would result in a worst case dynamic error of 0.08% at 100 cps for the 30K potentiometer. This is a factor of 5 improvement over the uncompensated 30K potentiometer. A few such two-tap potentiometers are recommended for even small computers.

3. Optimized-System Amplifier Frequency Characteristics

It is felt that some readers might be interested in seeing the advantages of capacitive compensation of Σ- amplifier feedback and input resistors through Bode plots rather than the root-locus of the text. Such data has been included for the Model 1048B Operational Amplifier as used in the EASE® 1100 Series analog computer. Since this gives a somewhat detailed view of the 1048B characteristics from a frequency basis, an equally detailed s-plane plot of the amplifier system open-loop characteristics has been included for those more interested in the root-locus approach.

Figure A.3.0 is an s-plane plot of 1048B and 1148 operational amplifiers - 1100 system open-loop poles and zeros.

Figures A.3.1 through A.3.4 show the theoretical and experimental frequency characteristics of the 1048B for various capacitive loads.

This analysis includes all three terms of Equation 19 for the transfer characteristics. The $\left[1 - \frac{Y_f}{G(s)Y_o}\right]$ term has the interesting effect of introducing a right half-plane zero.

Note that the 3-Σ amplifier loop gain margin can be predicted by tripling the db margin at 60 degrees phaseshift.

Figures A.3.5 and A.3.6 are a replot of the gain and phase for the same capacitive load data. Note that it takes a 0.01 capacitive load to achieve complex closed loop roots that are dominant.

Figures A.3.7 and A.3.8 show the bandwidth of a gain of 10 summer. Note that the stability margin remains excellent and the bandwidth is only slightly reduced.

Figures A.3.9 and A.3.10 show the effect of increased summing junction capacitance.

Figure A.3.11 shows that wider bandwidth can be achieved with larger than 43 pf capacity compensation. The data is for 100 pf compensation.

For a general purpose machine, 100 pf compensation is not recommended since 1) the stability margin is considerably reduced for the amplifier and 2) 1000 pf capacitors would be required for X10 input resistors. This would cause large coefficient pot phase errors.

Figure A.3.12 is included for a rough comparison of bandwidth when the system used uncompensated input resistors and a 10 pf feedback shunting capacitor. This characteristic is typical of many existing computers. For many standard problems it is adequate. However, the stability margin is much less, as typified by 3-Σ amplifier oscillation at a loop gain of 1.85, and Σ amplifier oscillation with 0.015 μf capacitive load. Further, the phase error without a 10 pf input capacitor is 0.36^o at 100 cps or 0.63% dynamic error.

References

1. Korn, G.A. and Korn, T.M., "Electronic Analog Computers", 2nd Edition, 1956. McGraw-Hill.

2. MacNeed, A.B., "Some Limitations on Accuracy of Electronic Differential-equation Solvers", Proc. IRE 40:303, 1950.

3. Marsocci, V.A., "An Error Analysis of Electronic Analog Computers", Trans. IRE-PGEC, December, 1956.

4. Dow, P.C., "An Analysis of Certain Errors in Electronic Differential Analyzers, I-Bandwidth Limitations, II-Capacitors Dielectric Absorption", Trans. IRE-PGEC, December, 1957.

5. Miura, T. and Nagata, M., "Theoretical Considerations of Computing Errors of a Slow Type Electronic Analog Computer", Trans. IRE-PGEC, December, 1958.

6. Single, C.H., "Precision Components For Analog Computers", I.S.A. Convention Record, Paper #56-21-2, September, 1956.

7. Brussolo, J. and Single, C.H., "Transient Analysis of Capacitor Equivalent Circuit", BECKMAN/Berkeley Engineering Report CRD59-5.

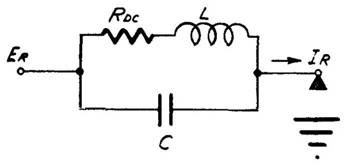
8. Meilander, W.C. and Hellman, B.H., "A Technique for Absolute Measurement of Analog Computer Capacitors", Goodyear Aircraft Technical Publication GER-9075, November, 1958.

9. Single, C.H. and Brussolo, J., "Copper-Mandrel Potentiometer Dynamic Error & Compensation", July, 1960 - Available from BECKMAN/Berkeley Division, Richmond, California.

10. Logan, B.C., "AC Performance and Phase Compensation of Copper-Mandrel Potentiometers", Technical Paper 497 - Available from BECKMAN/Helipot Division, Fullerton, California.

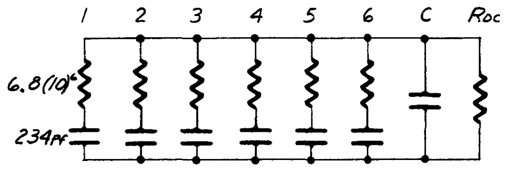
11. Schneider, F., Hiroaka and Gauldin, "Measurement and Correction of Phase-Shift in Copper-Mandrel Precision Potentiometers", Technical Paper 552 - Available from BECKMAN/Helipot Division, Fullerton, California.

12. Evans, W.R., "Control System Dynamics", McGraw-Hill, 1954



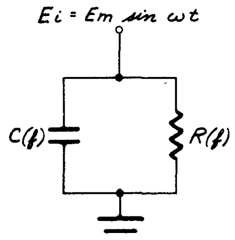
Resistor Equivalent Circuit

FIGURE 1



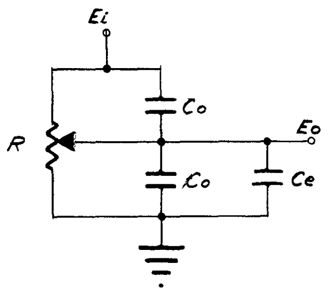
Fixed-Parameter Capacitor Equivalent Circuit

FIGURE 2



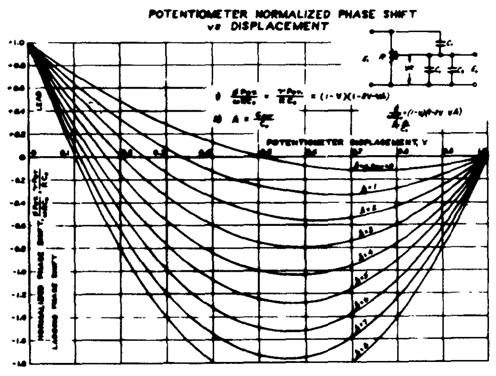
Frequency Dependent Capacitor Equivalent Circuit

FIGURE 3



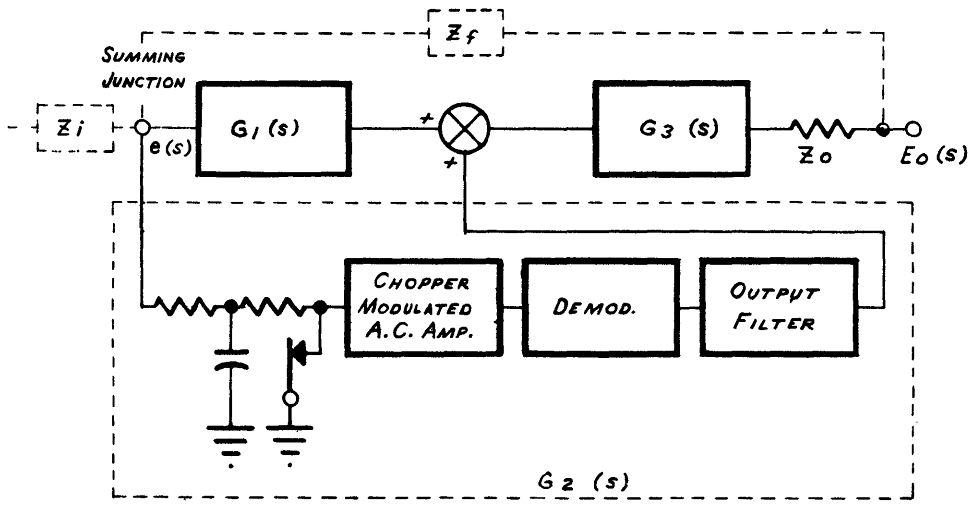
Potentiometer Equivalent Circuit

FIGURE 4



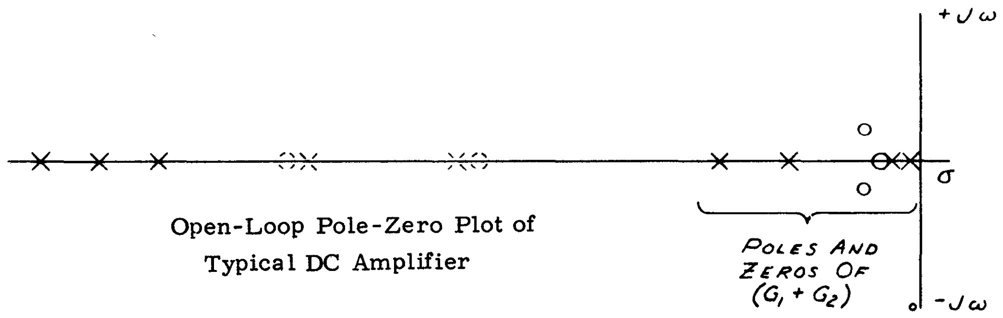
Potentiometer Normalized
Phase vs. Displacement

FIGURE 5



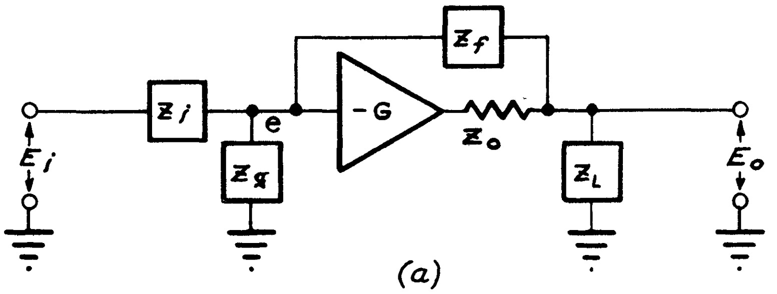
DC Amplifier Block Diagram

FIGURE 6



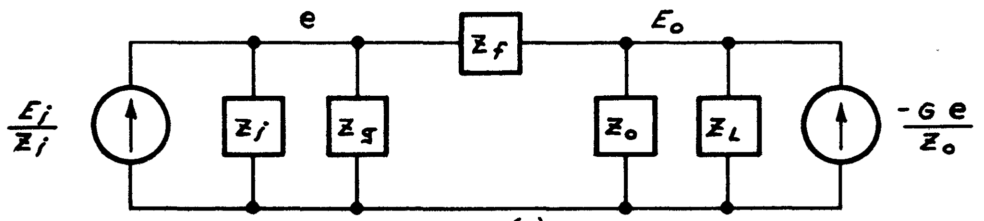
Open-Loop Pole-Zero Plot of
Typical DC Amplifier

Figure 7



(a)

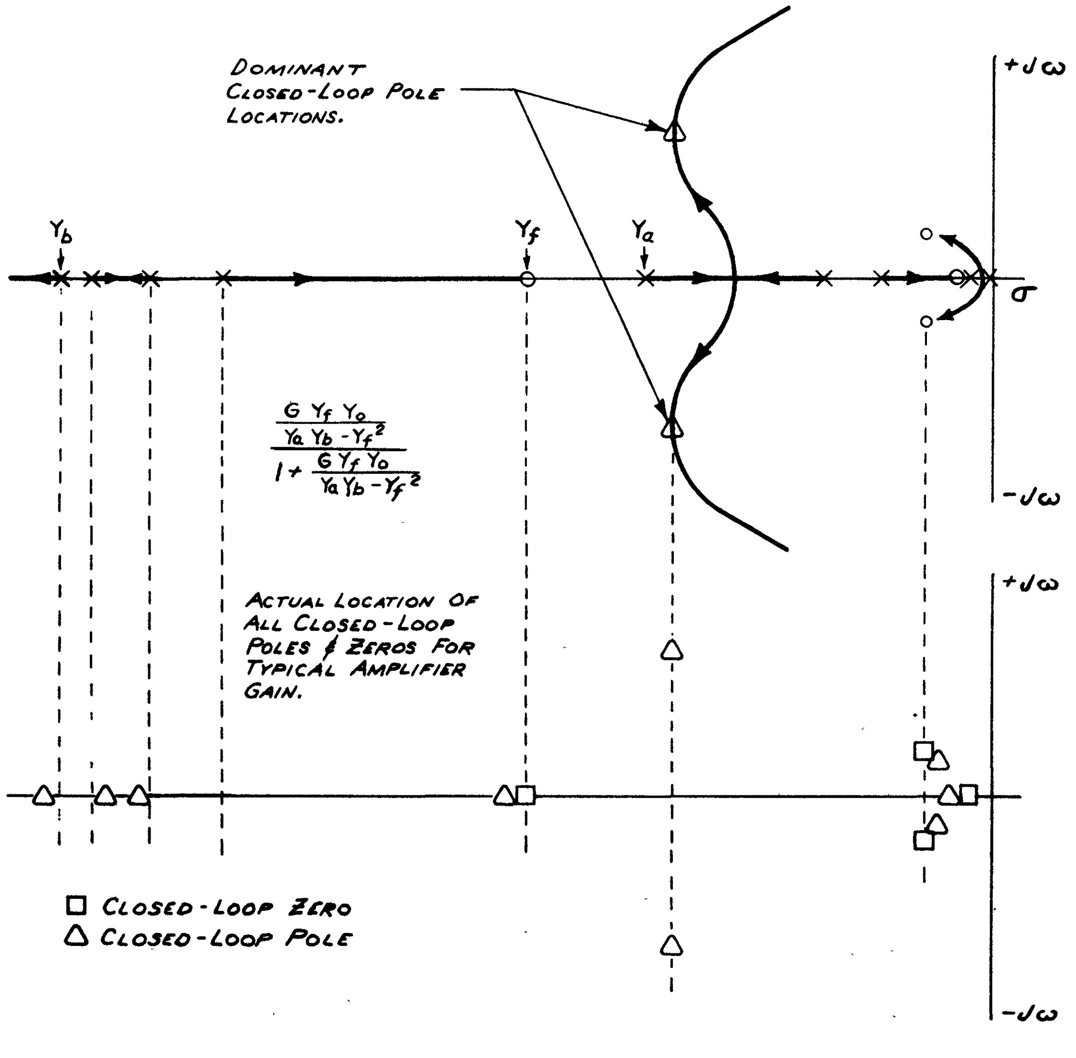
Block Diagram of Operational Amplifier in System



(b)

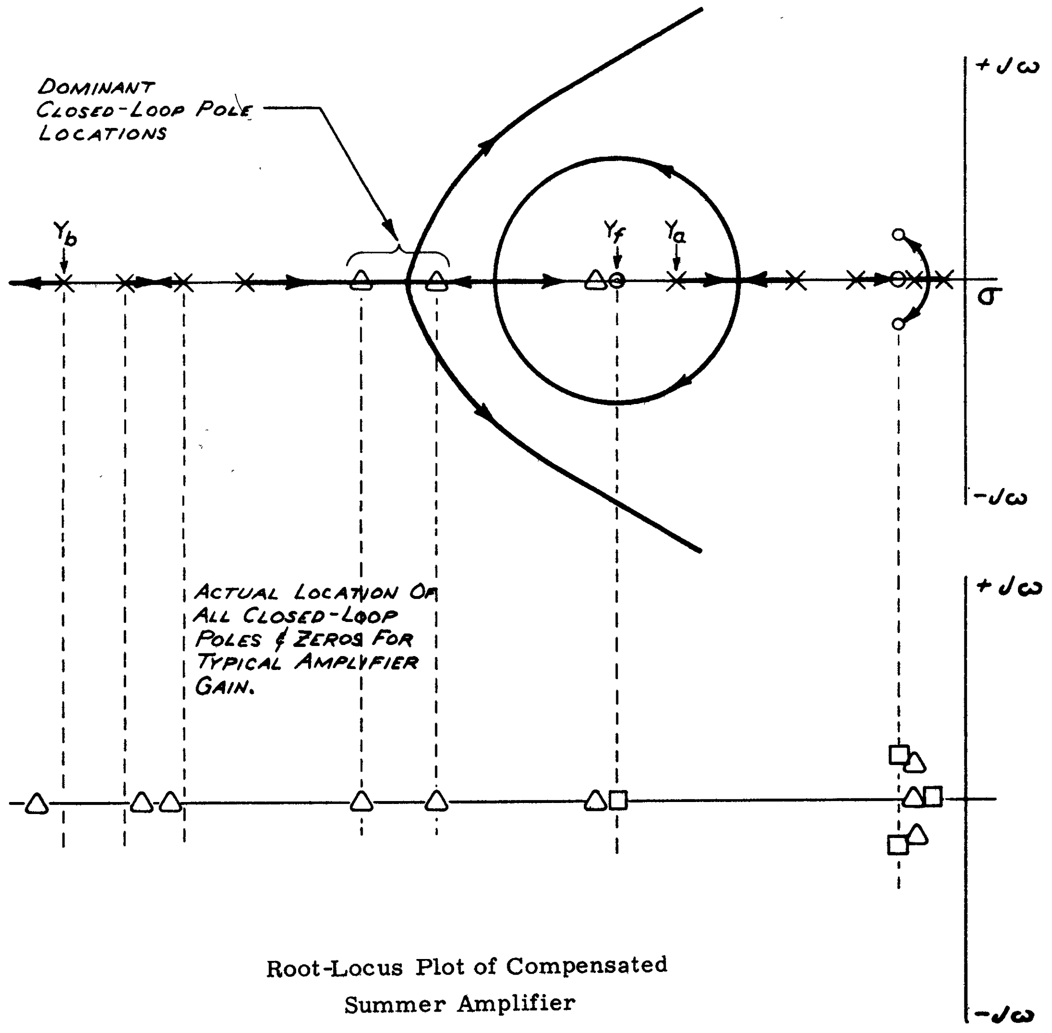
Operational Amplifier Equivalent Circuit

FIGURE 8


Dominant Closed-Loop Pole Locations.
Y_b
Y_f
Y_a
$+j\omega$
σ
$-j\omega$
$$\frac{\frac{G Y_f Y_o}{Y_a Y_b - Y_f^2}}{1+\frac{G Y_f Y_o}{Y_a Y_b - Y_f^2}}$$
Actual Location Of All Closed-Loop Poles & Zeros For Typical Amplifier Gain.
$+j\omega$
$-j\omega$
□ Closed-Loop Zero
△ Closed-Loop Pole

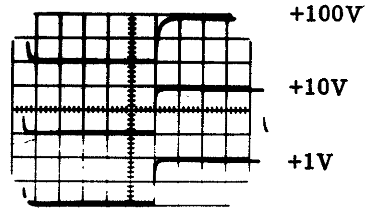
Root-Locus Plot of Uncompensated
Summer Amplifier

FIGURE 9


DOMINANT CLOSED-LOOP POLE LOCATIONS
Y_b
Y_f
Y_a
$+J\omega$
$-J\omega$
σ
ACTUAL LOCATION OF ALL CLOSED-LOOP POLES & ZEROS FOR TYPICAL AMPLYFIER GAIN.
$+J\omega$
$-J\omega$

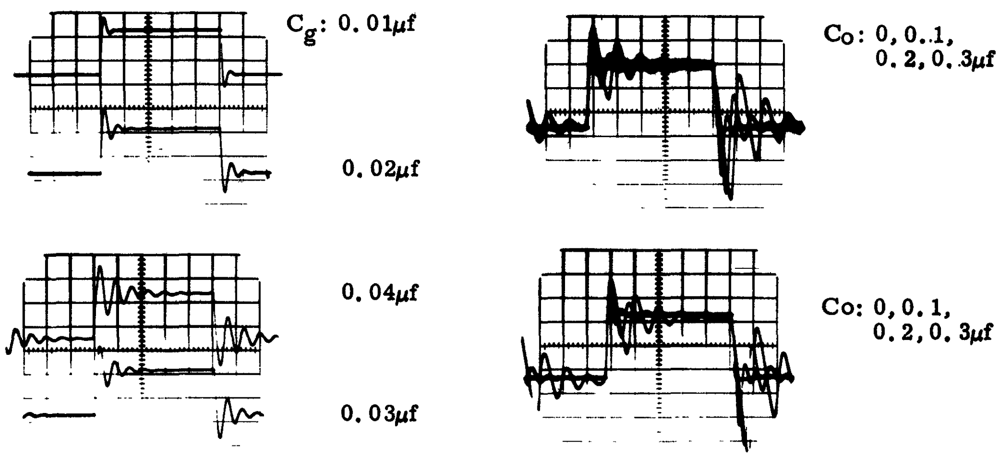
Root-Locus Plot of Compensated
Summer Amplifier

FIGURE 10



Step-Response of Compensated
Summer Amplifier

FIGURE 11



Effect of Summing Junction and Load Capacity on Step-Response of Compensated Summer Amplifier

FIGURE 12

MODEL 1148 STEP RESPONSE
THREE AMPLIFIER LOOPS AT VARIOUS GAINS AND
n-AMPLIFIER LOOPS AT UNITY GAIN

n-Amplifier Loop with Unity Inverters
Input: +10V, 400 cps

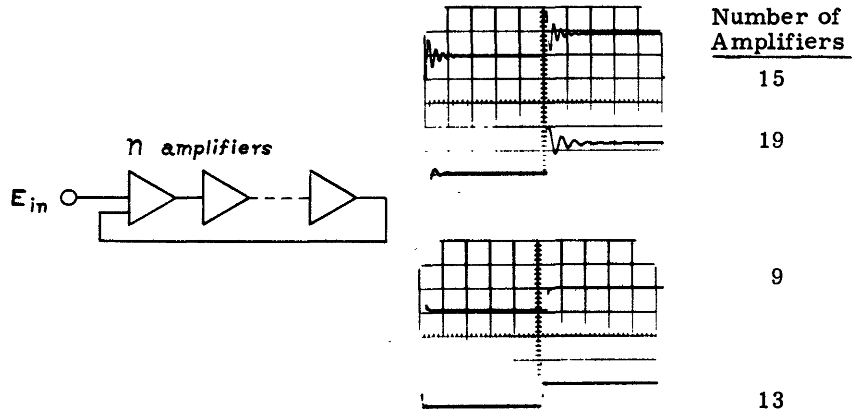


FIGURE 13

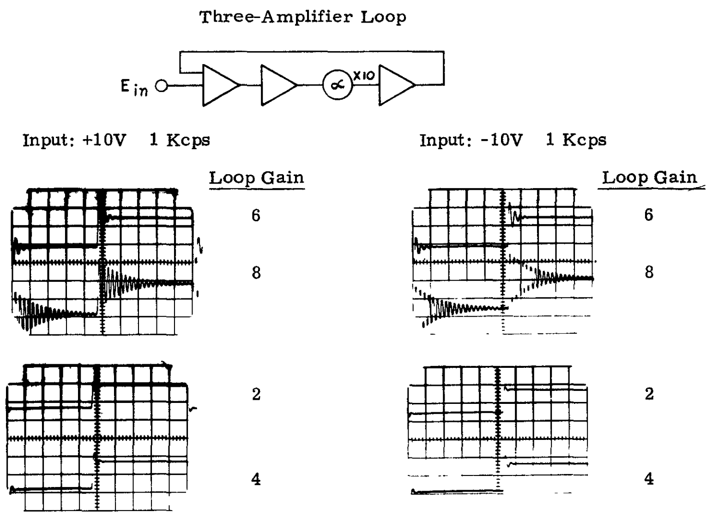


FIGURE 14

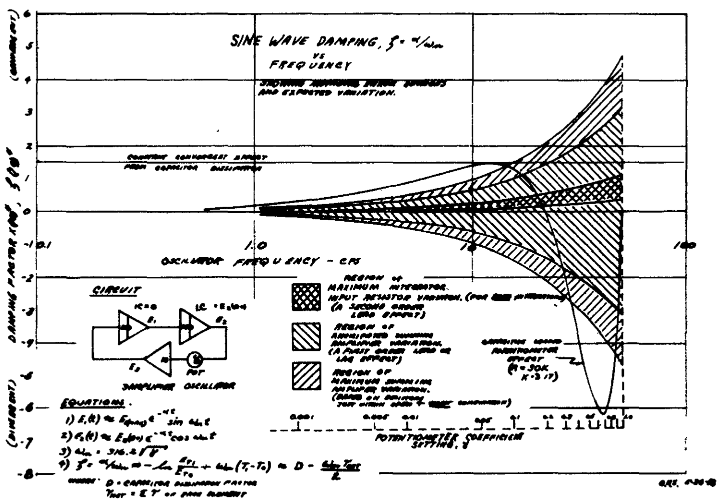


FIGURE 15

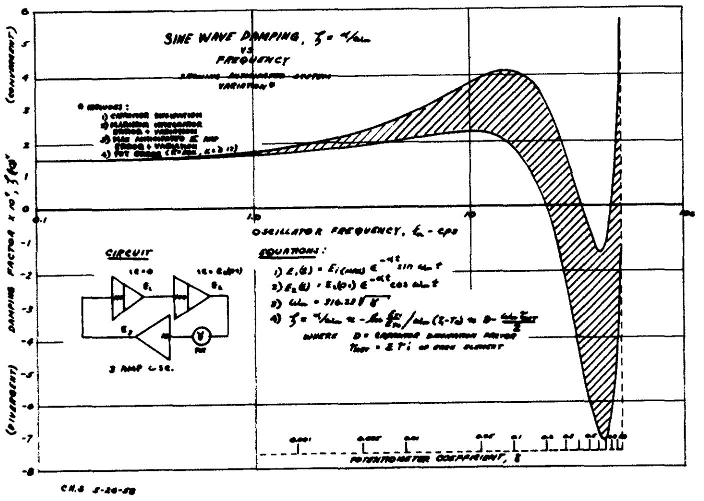


FIGURE 16

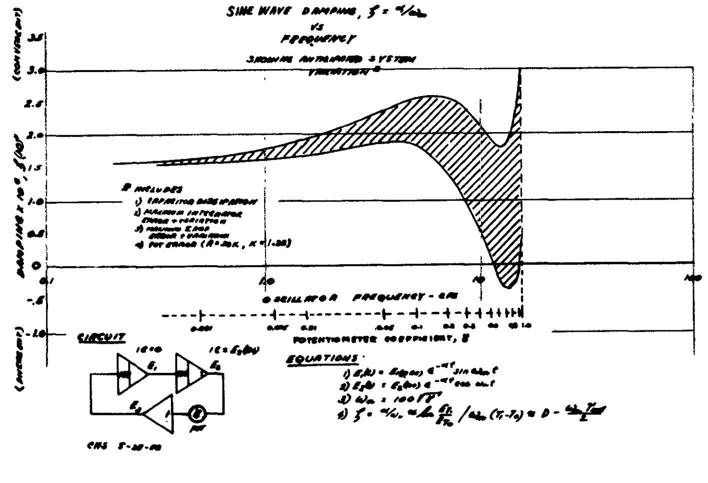


FIGURE 17

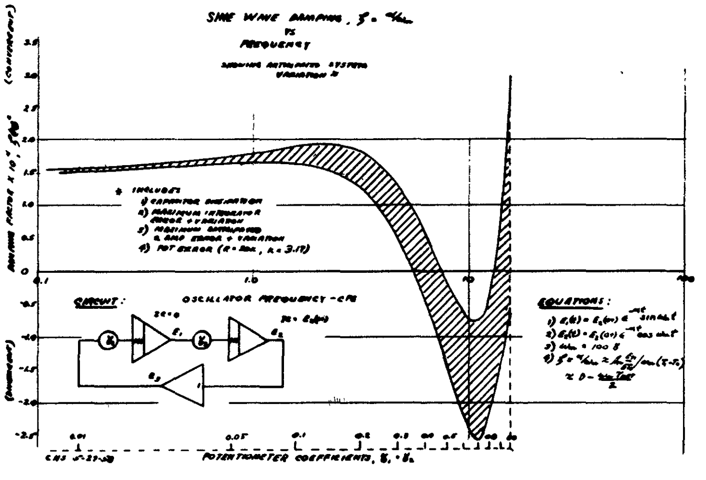


FIGURE 18

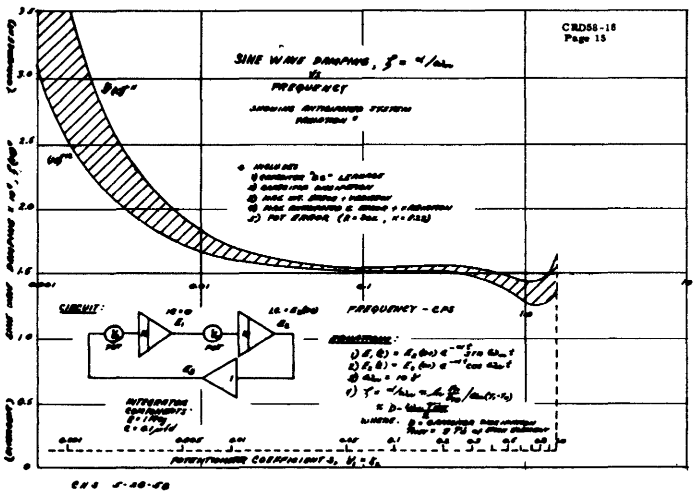


FIGURE A. 1. 0

Sine-Wave Damping vs. Frequency

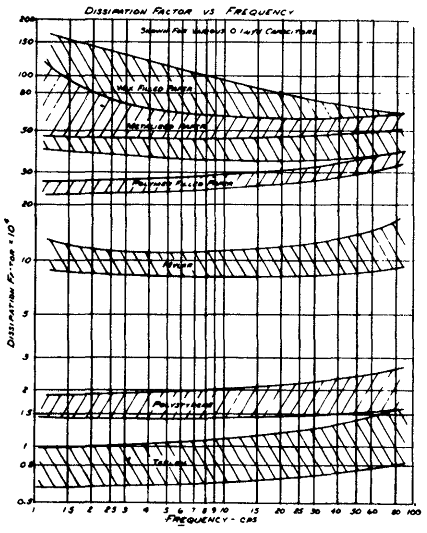


FIGURE A. 1. 1

Dissipation Factor vs. Frequency

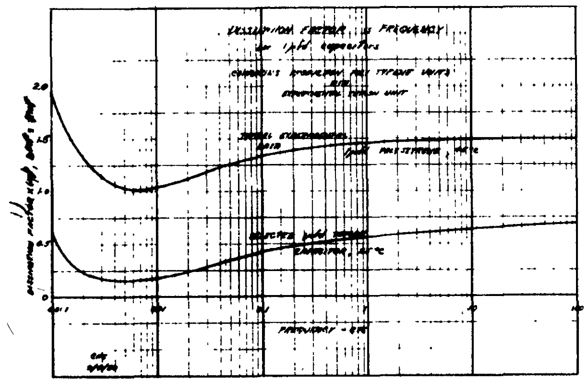


FIGURE A. 1. 2

Dissipation Factor vs. Frequency

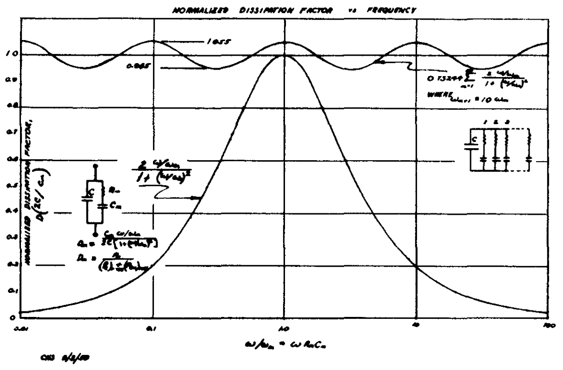


FIGURE A. 1. 3

Normalized Dissipation Factor vs. Frequency

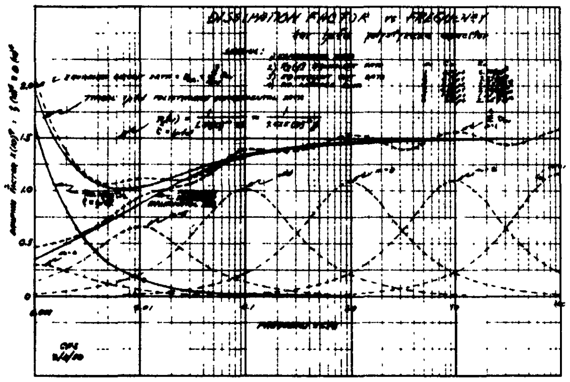


FIGURE A. 1. 4

Dissipation Factor vs. Frequency

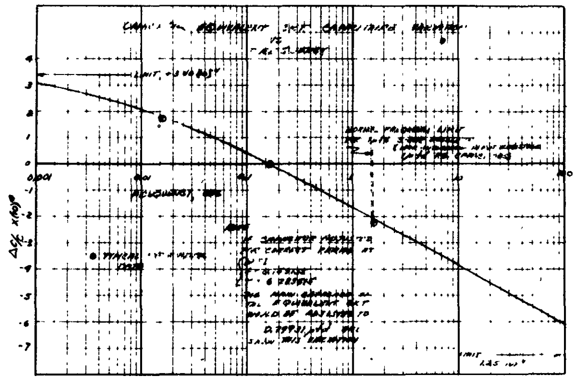


FIGURE A. 1. 5

Capacitor Variation vs. Frequency

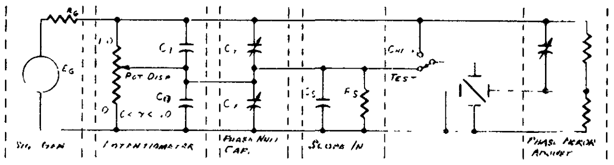


FIGURE A. 2. 0

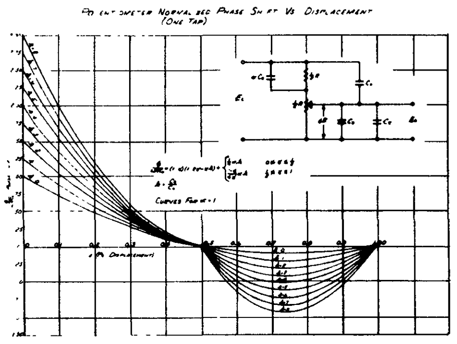


FIGURE A. 2. 1

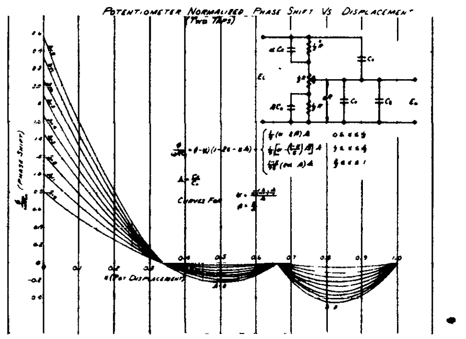


FIGURE A. 2. 2

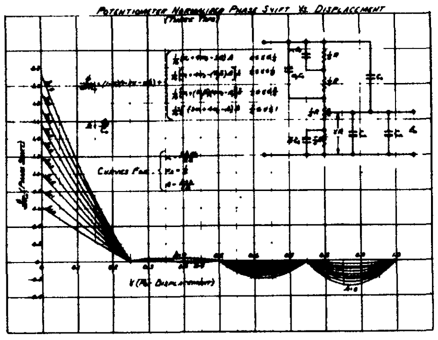


FIGURE A. 2. 3

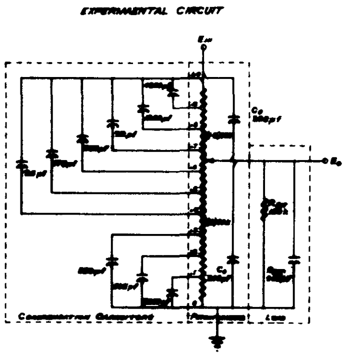
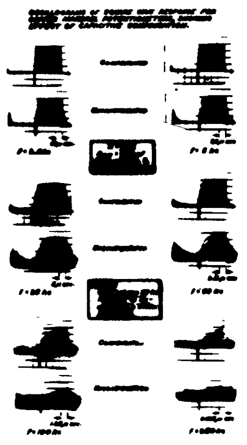


FIGURE A. 2. 4



See Next Page

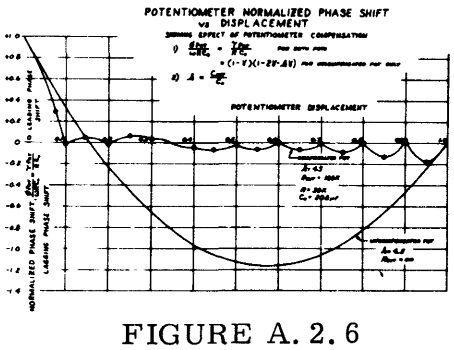


FIGURE A. 2. 6

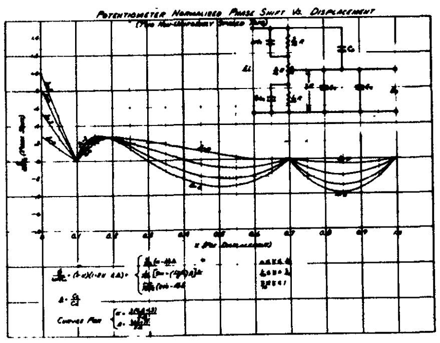


FIGURE A. 2. 7

FIG. A.2.5 OSCILLOGRAMS OF SQUARE WAVE RESPONSE FOR COPPER MANDREL POTENTIOMETERS, SHOWING EFFECT OF CAPACITIVE COMPENSATION.

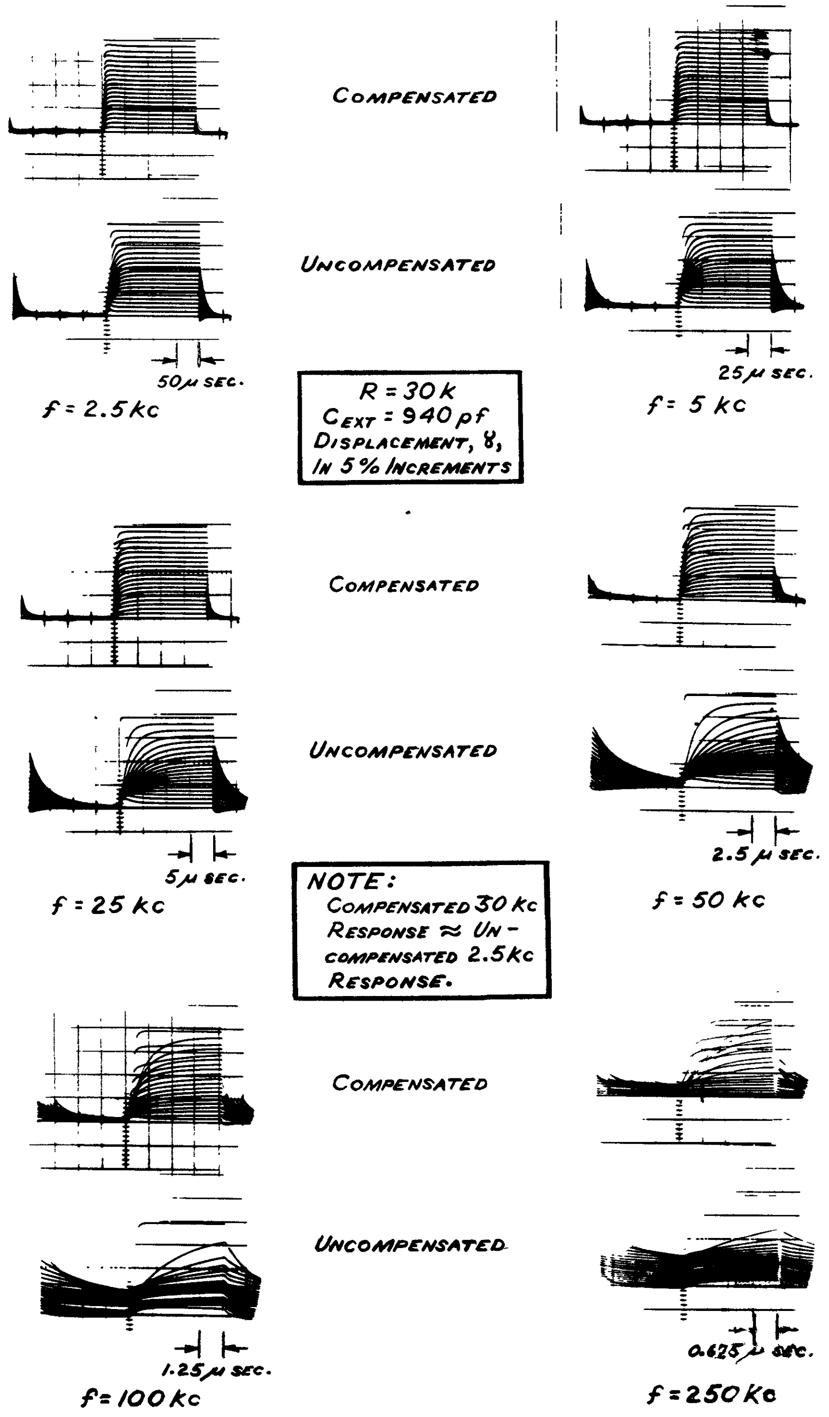
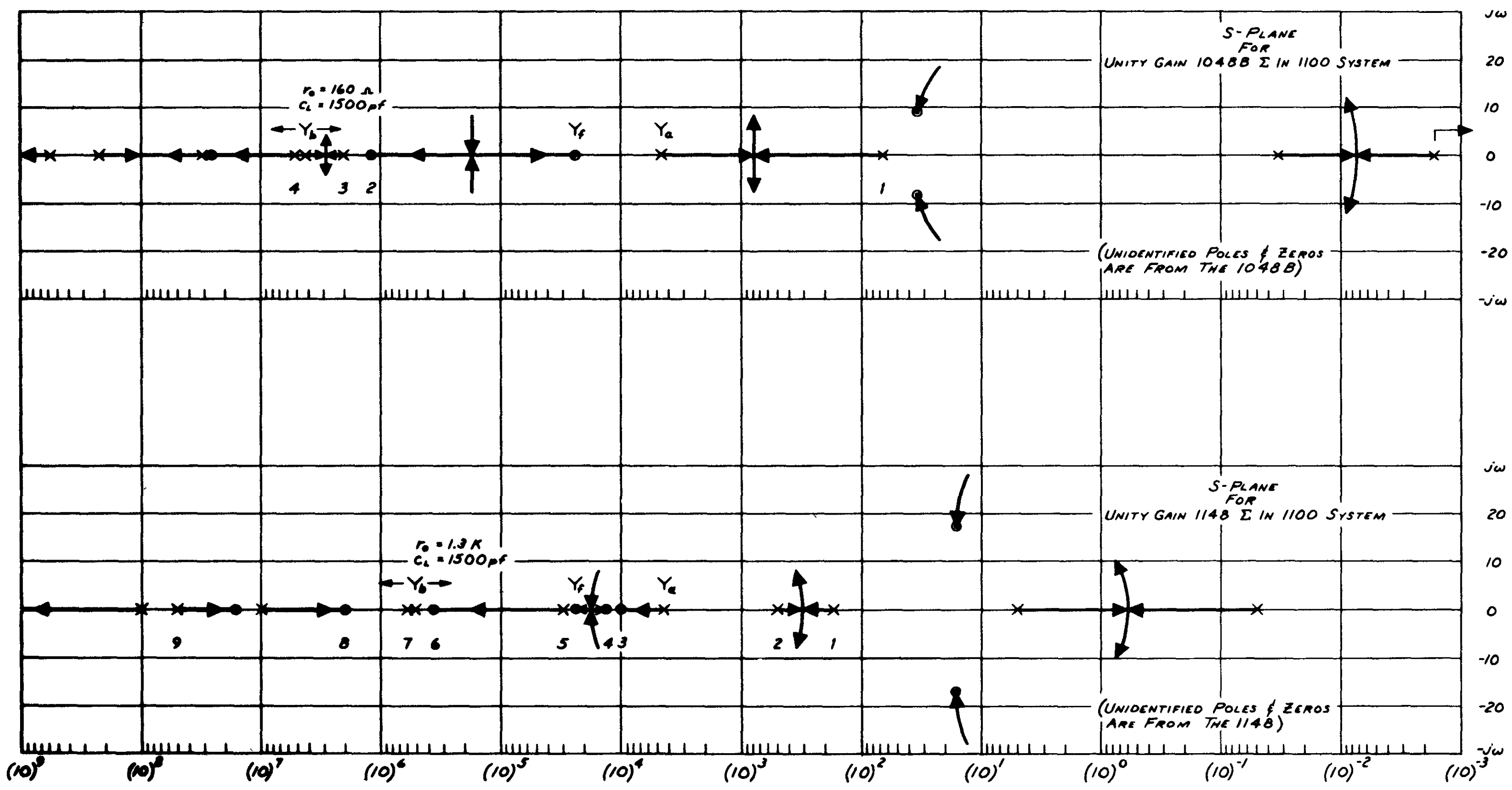


FIGURE A. 3.0

OPERATIONAL AMPLIFIER SYSTEM CHARACTERISTICS

SHOWING

OPEN LOOP POLES AND ZEROS
AND POSSIBLE ROOT LOCI
(NEAR THE REAL AXIS)



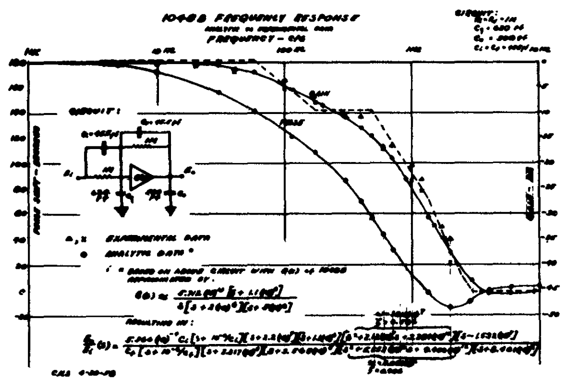


FIGURE A. 3. 1

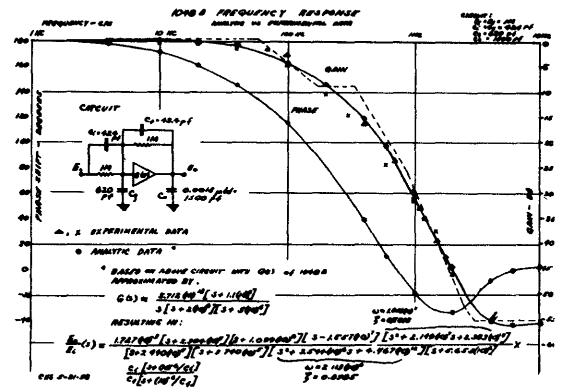


FIGURE A. 3. 2

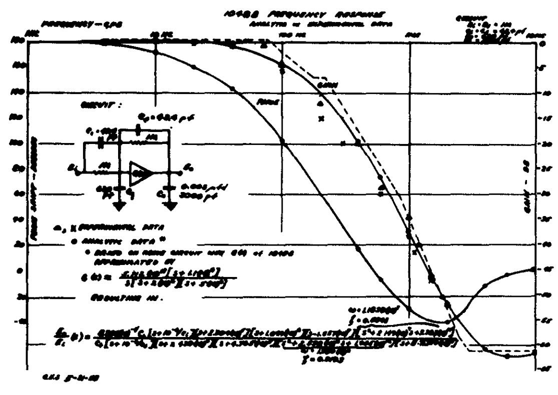


FIGURE A. 3. 3

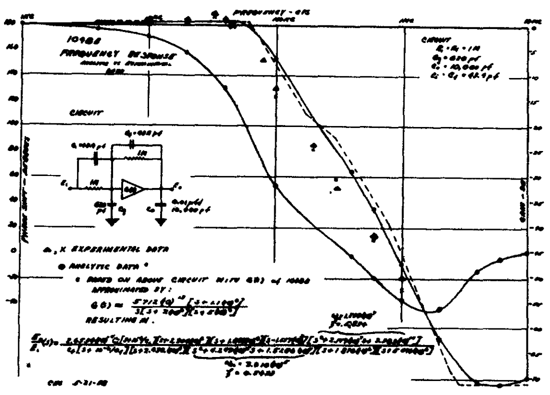


FIGURE A. 3. 4

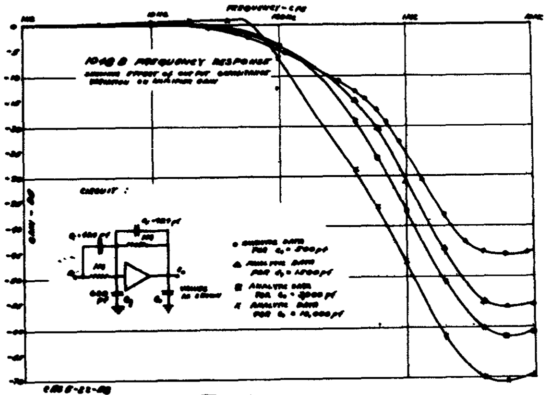


FIGURE A. 3. 5

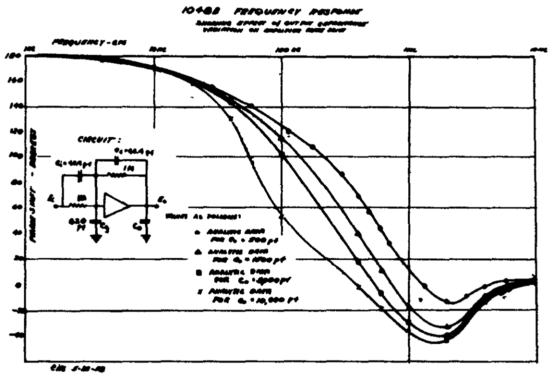


FIGURE A. 3. 6

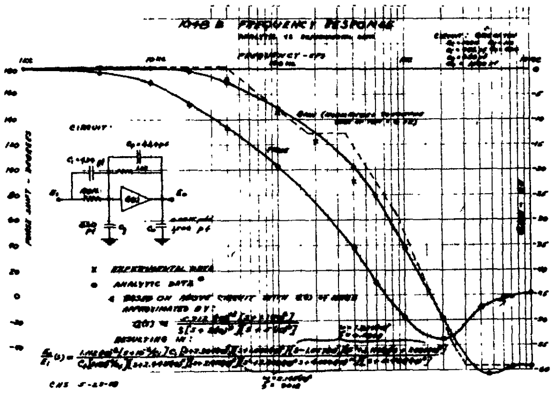


FIGURE A. 3. 7

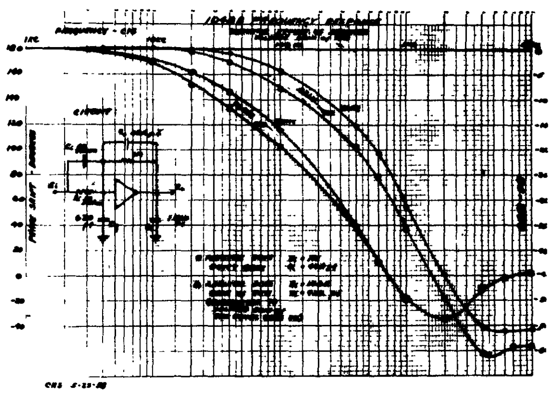


FIGURE A. 3. 8

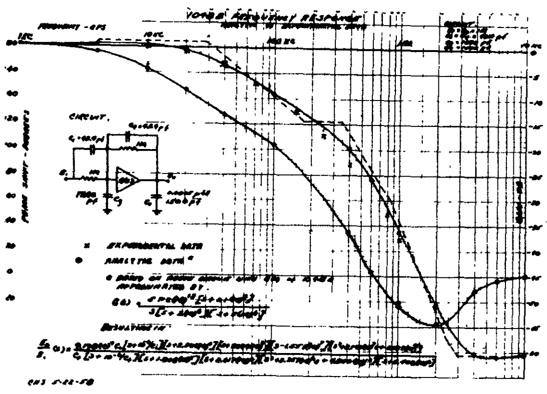


FIGURE A. 3. 9

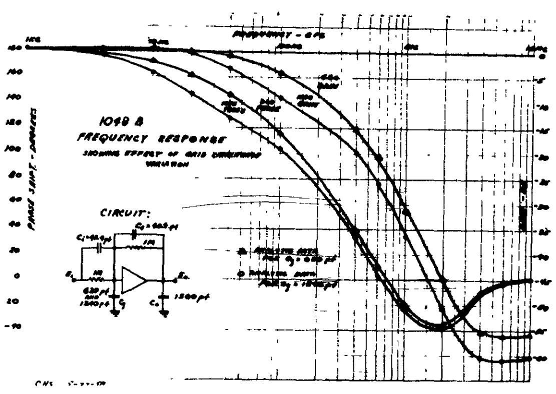


FIGURE A. 3. 10

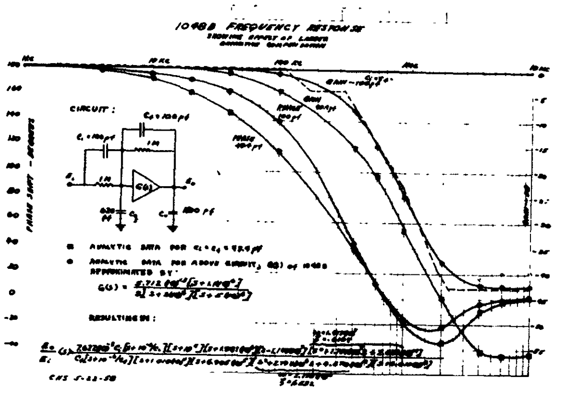


FIGURE A. 3. 11

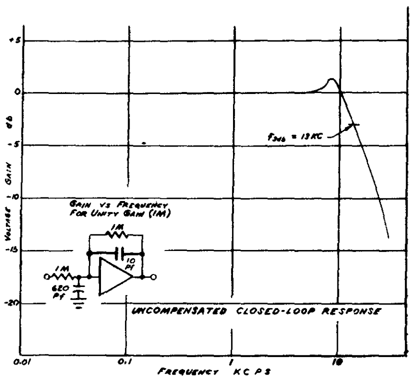


FIGURE A. 3. 12

DESIGN AND DEVELOPMENT OF A SAMPLED-DATA SIMULATOR

by

J. E. Reich and J. J. Perez
Members of the Technical Staff
Space Technology Laboratories, Inc., Los Angeles, California

Introduction

This paper describes the design and development of a sampled-data simulator (a special purpose analog computer) constructed recently at Space Technology Laboratories, Inc. (STL). The device was developed to simulate missile and spacecraft control system problems containing both continuous and sampled information. The machine has increased the speed of simulation, and decreased costs of operation.

Recent developments in space technology requirements, particularly of systems containing sampled or discrete data, present a class of problems requiring simulation of both sampled and continuous information containing both high and low frequencies. To accomplish this type of simulation on an analog computer, the sampled-data circuits which are described here were devised. The sampled-data channel consists of two zero-hold circuits in cascade, the first amplifier sampling the continuous input signal and the second amplifier presenting the stored information. Performance accuracy, which is within 0.01 percent of the desired value, was achieved by employing the special design and packaging techniques described.

Background

Sample-and-hold techniques have been in use on analog computers for a number of years. Several papers[1, 2, 3] report on the use of operational amplifiers and relays for the simulation of sampled-data systems. The same method has been used for sample-and-hold operation in generalized analog integration.[4]

Of the various methods used to simulate the sampled-data system with analog equipment, three had previously been exploited at STL, all with significant drawbacks.

The first method used a digital computer for the sample-and-hold operation and the solution of difference equations. It employed a combined simulation system comprised of an analog computer, the Univac 1103A digital computer, and the Addaverter analog-to-digital and digital-to-analog converter. This method was costly for problems requiring only a small amount of digital computation.

In a second method, the Addaverter was used by itself as a sample-and-hold simulator, with A-D and D-A channels connected in series to obtain and store the present and past values of a periodically sampled variable from the analog computer.[5] This method was time-consuming and employed an excessively complex device for relatively simple operations.

A third method used analog computer amplifiers and relays wired on a standard patchboard as sample-hold circuits. Although the method was inexpensive, the amplifiers and sampling relays did not have performance characteristics that made high-speed sampling possible and the number of components needed for such circuits was excessive.

Accordingly, investigations were made of methods to overcome the disadvantages of using analog computer components for sample-hold circuits. It was determined that by limiting the maximum sampling frequency to 100 cps, a high-speed relay could yield the desired performance. An investigation of the rise time and drift characteristics of currently available amplifiers showed that components were available with adequate specifications for a sample-and-hold circuit operating in the calculated frequency range of 0.05 to 100 cps.

On the basis of these studies, a prototype unit was fabricated and proved sufficiently successful in actual problem solution to justify the construction of the full-scale simulator.

Description of Basic System

Sampled-Data Channel

The principal requirement for a sampled-data channel with zero-order hold is to sample a continuous input signal periodically so that its output signal changes magnitude in a stair-step manner at the sampling instants, holding each sampled value for the period T. Figure 1 compares a continuous signal to a sampled-data signal and shows the pulse train which drives the sampling circuit.

The approach taken to satisfy this requirement was to use an analog computer operational amplifier, passive elements, and a high-speed relay connected as the sample-hold circuit. (Figure 2)

The circuit becomes a first order lag when the relay contacts are closed and the output charges to the amplitude of the input in an exponential manner. The lag time constant is equal to the value of RC. With the relay contacts open the amplifier is an integrator with no input $E_{in}(t)$, and the output $E_o(t)$ is equal to the charge on the capacitor C. The circuit will hold the charge until the relay is again operated and the capacitor is charged to a new voltage level. The RC value can be minimized until equal to the amplifier rise time, which was chosen to be a factor of 10 smaller than the time the relay contacts are held closed.

It was necessary to have a means whereby the basic channel could be connected to other channels and the signal delayed up to six sampling periods. Rather than using cams and switches[6] to sequentially drive the sample relays, two sample-hold circuits of Figure 2 were cascaded to give the sample-and-hold or sampled-data channel shown in Figure 3.

In the sampled-data channel, the Sample amplifier output is stored in the Present amplifier circuit to prevent loss of the information when the next Sample pulse operates the Sample relay. Transfer of the signal is accomplished by delaying the Sample pulse to the Present relay by a period having a range from the minimum time the relay is closed (a factor of 10 greater than the RC value) to the maximum of one second, the maximum time being determined by the delay-flop (monostable multivibrator) design. This method permits cascading of channels to obtain a Sample period delay per channel plus a small Present delay. Noticeable effects do not occur in most problem simulations when the delay is set at its minimum value.

System Controls

To operate two groups of channels at different related frequencies, the groups running at the higher frequency are operated as described, but the input for the lower frequency group must be obtained by dividing the higher frequency f_o by some integral number.

To instrument difference equations directly at a patchboard, summing amplifiers and coefficient potentiometers are required. Mode controls for the summing amplifier relays and the sample-and-hold relays include potentiometer setting (Pot Set), initial conditions (IC) on Present amplifiers, sample-hold amplifier integrator hold (Hold), and the start of computer operation in synchronism with the first clock pulse (Compute). Such controls, with the digital logic required for pulsing the sampling relays, determined the basic system indicated in Figure 4.

In Figure 4, note that the remote analog computer can control or be controlled by the simulator as required. However, the simulator can be operated independently from its own control panel. The purpose of the Hold Synchronizer (start control) is to start the simulator in the Compute mode coincident with the first sample pulse after the operator has changed the mode switch to Compute. This prevents the occurrence of a fractional interval during the first sample period. All sample and present relays are energized to clear an initial input voltage just before the simulator is in the Compute mode by the automatic clear circuitry, or they can be energized at any time by depressing the master clear switch. The frequency divider (counter) reduces the frequency for the low frequency group pulse formers if so desired. The Clock is a low frequency square wave generator that is connected to the Hold Synchronizer. The output pulse width of the Hold Synchronizer is reshaped in the pulse unit before the relay driver and closes the sampling relay contacts for a period (10 RC time constants) that will insure the desired accuracy of the sampled signal.

Design Considerations

The major problems encountered in the design tasks were direct consequences of using both analog and digital components in the same system. Previous experiments with diode gating circuits as sampling devices in sample-hold circuits had proved that these diode gate circuits introduce excessive noise into the amplifiers, that "open" impedance is too low, and that a well-regulated power supply is required for biasing the diode gate.

Therefore, objectives of sampling relay circuit design were to select a high-speed relay, and to design (a) a compatible relay driver circuit, (b) a pulse-former circuit to shape the width of the sampling pulse, (c) a present delay circuit to operate the present relay after the sample relay is actuated, and (d) pulse control circuits to start the operation of sampling relays with the analog computer control mode relays.

Sampling Relay Circuitry

Tests of a chopper as a sampling relay in a sample-hold circuit showed that the relay operated up to 1200 cps without any noticeable effects in the dwell time or phase-lag. The relay is a polarized single-pole, double-throw, non-resonant switch that provides break-before-make action in synchronism with the current-wave of the driving source. The mechanism is housed in a metal case having a plug-in header and captive, locking external shield. The unit fits a standard skirted 7-pin socket.

Electrical performance and life of contacts are closely related to the characteristics of the circuit into which the relay is used. Standard contact-rating is 3 volts, 2 milliamperes, resistive load. Contact voltage could be restricted by using a diode limit circuit on the amplifier grid.

To drive the relay at the sampling frequency of the clock source, a driver circuit (Figure 5) was designed. The circuit is basically transistorized switches that operate from the output of a pulse-former. A negative pulse causes Q1 to conduct, the current is forced from ground up through the coil, and the contacts (6 and 7) are closed for the duration of the sampling pulse. A positive pulse forces current in the opposite direction, closing contacts 1 and 7.

In order to provide sufficient time for the capacitor in the RC network to charge to within 0.01 percent of the sampled input level, the relay contacts must be closed for a duration of time greater than the RC value. The pulse-former circuit shown in Figure 6 is used to turn on the relay driver (closing the relay contacts). The delay-flop provides the proper pulse width to hold the relay contacts closed for 5 percent of the maximum sampling period, and its voltage levels are changed by Q_3 and Q_4 to operate the relay driver stage.

To have a variable delay of the present pulse, a delay-flop with an external adjustable capacitor is used. The capacitor value is determined by

$$C_x = 50(t - 2)$$

where C_x is the external capacitor in micro-microfarads, and t is the delay time in microseconds. A total of 50 microfarads capacitance may be used to give a maximum delay of approximately one second. The unit is followed by a blocking oscillator that works into the pulse-former of the present relay.

In the pulse control circuits now to be discussed, the interrelations of the various blocks determine the typical impedance levels, gains, etc.

A hold synchronizer circuit is used to change the signal levels from the clock source to a level suitable for operating the pulse units. The circuit has a transistor binary switch that supplies voltage to the computer and/or simulator Hold relays for a time after the operator has put the mode control switch to the Compute position. The synchronizer is shown in block diagram form in Figure 7.

The clock input to the Hold Synchronizer is in the form of a square wave (frequency selected by the operator) and is supplied to the synchronizer delay-flop. In order to synchronize the operation of the computer and the first pulse to the sampling relays it is necessary to allow for the "flip-time" of the binary switch. This is accomplished by delaying all pulses to the "and" gate. Two blocking oscillators are used to maintain the rise time and logic levels of the transmitted pulse, one before and one after the "and" gate. Pulses out of the delay circuit are also amplified and applied to the transistor binary switch, which is essentially by-passed by the computer (or simulator) mode control bus when the mode switch is in any position other than Compute.

When the mode control switch is first turned to Compute, voltage is removed from the Compute bus, thereby enabling the binary switch to operate the Hold relays. As the next pulse appears at the input of the binary switch, the "and" gate is enabled and the voltage from the Hold bus of the computer or simulator is removed. Thus, coincidence occurs between the Hold Synchronizer pulse train and the Compute mode in precise synchronism.

A frequency divider is used to divide the pulse frequency from the Hold Synchronizer to the pulse-formers for the sample and present relays. Each of the two ten-point decade counting units has a selector knob allowing the set-in of any desired value from 0 to 9, the actual number chosen being indicated by the selector knob dial. After presetting, the coincidence output signal is generated whenever the input signal count agrees exactly with the chosen preset number. The two units permit a total count and preset number range from 0 to 99.

A clearing circuit consists essentially of a control relay whose contacts apply voltage to the relay driver units, causing the sampling relays to sample. This sequence clears the input to each sample amplifier and presents any IC value on the output of the present amplifier. The clearing circuit is actuated automatically or by a manual Master Clear switch.

Two-Channel Prototype

A prototype unit was constructed after the circuits were developed and the relay selected. The unit contained sampling relays, passive networks, control relays (Pot Set, IC), Hold Synchronizer, frequency divider (counter), and digital logic. Four amplifiers were cabled from the analog computer to the unit for the two sampled-data channels. The prototype unit was used successfully in many problem simulation studies.

Amplifier Selection

The prototype unit sampling frequency was limited by the amplifier performance characteristics. Amplifiers made by various analog computer manufacturers were tested, and an operational amplifier eventually was selected for sample-hold applications. In addition to the general requirements of output current, gain, frequency response, and phase shift, it was found that the amplifier integrator drift during a hold time of 4 seconds and with a 0.001 microfarad feedback capacitor is 2 millivolts, that its noise is 5 to 10 millivolts, and that amplifier rise time is 40 microseconds.

Simulator Specifications

At this point simulator specifications were formulated upon performance of the previously designed circuits, high-speed relays, and the selected amplifier, after which development was initiated.

The simulator has 12 sampled-data channels, each consisting of two amplifiers in cascade. The first amplifier samples on command of a Sample pulse. The second amplifier will sample on command of a Present pulse. In addition to the sampled-data channels, 12 summing amplifiers and 20 coefficient potentiometers are available to make possible the instrumentation of difference equations directly on the simulator patchboard.

The simulator is a self-contained unit with its own power and relay supplies, controls, and patchboard. An insulated patchboard was used for inputs and outputs of system components.

A 0.001 microfarad capacitor is employed in the sample and present amplifier feedback. The maximum allowable drift specified was 2 millivolts in 4 seconds, or a rate of 0.5 millivolt/second. The noise level measured at the patchboard is less than 10 millivolts, a good value since the patchboard is not shielded.

Logic elements are transistor type with inputs and outputs brought out to the patchboard. These elements, with logic levels of -3 and -11 volts, include flip-flops, delay-flops, "and" gates, "or" gates, emitter followers, blocking oscillators, Schmidt triggers, and level triggers. The flip-flops, delay-flops, and blocking oscillators trigger on 5.5 volts in at least one microsecond. A falling waveform will not trigger any of these circuits, regardless of voltage or slope of signal.

The maximum and minimum delay of the present pulse are one second, and 500 microseconds respectively; a delay-flop is employed. Adjustments for varying the delay time are placed on the left side of the patchboard panel, one for each of the four present pulse former units. External capacitors may be added by means of banana jackets located on the chassis housing the pulse former units, accessible from the rear of the control cabinet.

Amplifier Circuit

Each amplifier grid, output, and overload wire is cabled to the control cabinet for connection with passive networks, operational and sampling relays, and to indicate an overloaded amplifier at the central overload indicator on the control panel. The plate and filament voltages are cabled to the amplifier chassis within the amplifier cabinet. The summing amplifier networks have two 100K ohm and two 1 megohm input resistors with a 1 megohm feedback resistor. The sampling amplifier networks have four 50K ohm input resistors and the present amplifier has one 50K ohm input resistor.

To indicate that an amplifier is in an overload condition, a light on the control panel is turned on and an audio alarm is actuated. The alarm has both tone (frequency) and volume adjustments.

Simulator Control Circuits

Control panel switches operate relays that turn on or off the filament, plate, and reference voltages for the simulators. As the switch is pressed a light indicates the "on" condition.

In order to read out the voltage of power supplies, trunk lines, potentiometer arms, and the sample, present, and summing amplifier outputs, a pushbutton selector system was designed. A selector switch of 100 points and a relay of 50 points connected in series form a 150 point system. Points 1-99 are selected by the switch and points 100-150 are selected by the addition of the relay.

The mode controls are relays operated by control panel switches. The operating mode is indicated by a light on the surface of each switch.

Simulator Development

Patchboard Layout

The patchboard is an insulated, 816-hole board. Colors were applied to the board by an inexpensive photo-emulsion process rather than by the standard but expensive silk-screening method. Colors denote inputs, outputs, ground, etc. All inputs, outputs, trunk lines, and multiple points are numbered for clear identification (Figure 8).

Amplifier Installation

All amplifier chassis are mounted in the amplifier cabinet (number 23 shown on the right side of Figure 9).

Amplifiers are packaged in groups of four (or quad) and three groups are mounted in one amplifier chassis. Figure 9 shows the following system components in the amplifier cabinet (from top down): 12 Sample amplifiers, 12 Present amplifiers, digital voltmeter, clock, filament transformers (behind shelf), 12 Summing amplifiers, and plate supply unit.

Oven Fabrication

It was necessary to locate the oven as near as possible to the patchboard (as shown in the control cabinet in Figure 9) to minimize noise pickup. An oven thermostat, having a set point of 100°F, controls the temperature environment for the passive networks. Heat is generated by power resistors and the air is circulated by a small blower mounted on the back of the oven. Capacitors are adjustable through access holes in the front of the oven. The connections for the plug-in networks are also mounted on the front of the oven to facilitate wiring.

Potentiometer Panel

Twenty coefficient, hand-set potentiometers with input switches and arm fuses were assembled on a panel and mounted directly above the patchboard panel. Pots are ten-turn Helipots, with 30K ohm resistance and 0.1 percent linearity.

Control Panel

Figure 10 shows the control and patchboard panel. On each side of the patchboard are dials and switches for selecting the capacitor value for the present pulse unit delay (left side) and spare delay-flops (right side). The following controls are shown on the control panel: power-controls, flip-flop clear lights, divider (counter) clear light with frequency indicator lights and selector knobs, function switches, amplifier overload lights, readout (address) selector pushbuttons, Level Trigger polarity selection knobs, overload test switch, master clear switch, switch for connecting digital voltmeter to address selector or patchboard terminal, and control mode switches for Slave, Pot Set, IC, Hold, and Operate.

Wiring

The most time-consuming phase of the development program was the wiring of the simulator. Extreme care was taken to shield wires carrying d-c voltages from those carrying pulses and a-c power. A carefully designed ground system was used to prevent the introduction of a-c noise signals to the input of the high gain operational amplifiers in the simulator.

Signal Ground. The amplifier grid wire is shielded to prevent electrostatic induction of noise. The shield is used as the ground for only the wire it shields, and is tied to another shield only at one point where both are grounded. A copper bus bar is located behind the patchbay for connecting all types of grounds, except the a-c neutral. This bar is then connected to the facility earth ground by three No. 8 size wires to cause the bar to act as the earth ground of the simulator. In addition, the cathode input stage of the amplifier stabilizer has the cathode ground brought directly to the system earth ground to minimize interaction by amplifier channels.

Power Ground. All power currents return only through power ground leads. At no place is the power ground connected to the signal or chassis ground on the amplifiers. However, the power ground is connected to these grounds on the earth ground bus mentioned above.

Chassis Ground. The chassis of all units are grounded through the slides which hold them in the cabinet, and the panel screws. The cabinets are in turn grounded to the earth ground bus in the control cabinet. Each unit as well as the whole system is then in effect enclosed in a Faraday shield.

A-C (Neutral) Ground. The a-c ground, sometimes called an industrial ground or the a-c neutral, is the fourth wire of the three-phase power brought to the computers in the facility. To prevent a-c noise pickup in the equipment, this ground is not connected to any of the above grounds but serves only for the return of a-c current.

System Performance Data

Typical new system testing was performed before the simulator was put into operational use. However, some impossible-to-anticipate difficulties were not encountered until the machine was actually used for several different problem simulations. The data given in this section prove the degree of accuracy with which the system met specifications.

Test Problems

The most essential requirement of the sampled-data channel is that each channel cascaded to the first channel give a sample period delay. This was investigated by imposing a sinusoidal or triangular waveform at 0. 01 cps on the input and a sample period of 5 seconds to the first channel, and recording the input and outputs of the three channels cascaded (Figure 11). Although the amplitude of the sampled signal appeared identical as it passed through each cascaded channel, a special circuit was mechanized to detect the amplitude error in the output of each channel when six channels are cascaded. Each sampled-data output was compared to the input, with the error signal recorded. The input signal was 12 volts d-c. In Figure 12, the maximum error recorded is 20 millivolts in the output of the sixth channel, which indicates some of the error is accumulated.

Next, the output signal variation, with the input signal varied in frequency and the sampling frequency held constant, was investigated. A triangular and then a sinusoidal input signal of 100 volt amplitude was varied in frequency from 0. 05 to 50 cps in increments of 10 cps. The sampling frequency was constant at 100 cps. As viewed on an oscilloscope, no apparent change in the output waveform was noted.

Again, the input was a triangular waveform, but had a fixed frequency of 0. 05 cps. The sampling frequency was varied from 0. 01 to 100 cps in 10 cps increments. No apparent change was noticed in the output signal for this test.

Amplifier Characteristics

Integrator Drift. The integrator drift rate is of prime importance during the lowest sampling frequency selected. To maintain the accuracy given in the specifications required the drift rate to be 0. 5 millivolt/sec during a sampling period of 4 seconds. In addition, the specifications required the maximum drift to be within 20 millivolts during the maximum sampling period, or 0. 01 percent of 200 volts full scale.

Integrator drift is due mainly to two factors: (a) the capacitor leakage in the circuit while holding or storing the sampled signal, and (b) the amplifier offset voltage caused by input-tube grid current. Precautions were taken in wiring the simulator to minimize possible capacitor leakages in the system.

The measured drift rate of the system met the required specification only after careful adjustment. However, a drift rate of 3 millivolts/second, which could be obtained with relative ease, was used to determine the lowest sampling rate.

Noise Level. The output noise level has a high frequency content well within 10 millivolts peak-to-peak, while the low frequency is within 5 millivolts peak-to-peak. In initial bench tests of the circuit, the total noise level was within 4 millivolts.

Transient Response. It was desirable that the amplifier transient response have a rise time of 50 microseconds. The rise time, measured from 10 to 90 percent of the final value, was found to be 50 microseconds. This satisfies the specification that the amplifier rise time be equal or less than the RC network time constant.

Crosstalk and Random Triggering

The cross talk measured at the patchboard is only 20 millivolts when the output of one amplifier is 100 sin ωt and the input of the measured amplifier is open.

Occasionally, while checking the performance of the simulator, fractional sampling would occur. An investigation revealed that this occurred whenever test equipment on the simulator a-c line was turned on or off. The source of the trouble was traced to the delay-flop. It was found that noise or transients would change the output state of these units, which are used throughout the system wherever a delay is desired. Since the Hold Synchronizer has a delay-flop the problem may be started prematurely, resulting in fractional interval of the first sample period. Functional sampling may also occur at any time during the problem as a result of a false or undesired pulse triggering the delay-flop in the pulse former units. In order to eliminate this problem, a redesign of the delay-flop is under consideration at present.

Conclusions

The results of performance evaluation tests indicate the lowest sampling frequency (within 0.01 percent accuracy) to be 0.125 cps when critical adjustments are not made. This is considerably higher than the calculated frequency of 0.05 cps. One method to lower the limit to 0.05 cps would be to lower the maximum sampling frequency. This would allow an increase in the value of the feedback capacitors, thus reducing the amplifier drift. The sampling pulse width would, of course, be correspondingly increased. Amplifier rise time could then exceed its previous maximum limit of 50 microseconds, since no advantage accrues from restricting it below the RC time constant of the network.

Acknowledgement

The authors wish to express their appreciation to R. Lunden, of the STL Analog Computation Center, for helpful ideas and supervision of packaging the simulator.

References

1. Wadel, L. B. "Analysis of Combined Sampled and Continuous Data Systems on an Electronic Analog Computer," IRE Convention Record, Pt. 4, pp 3-7, 1955.

2. Chestnut, H., A. Daubul, and D. Leiby "Analog Computer Study of Sampled Data Systems," Proceedings of the Conference on Computers in Control Systems, (Also General Electric Report No. 57GL351), October 1957.

3. Elgerd, O.I. "Analog Computer Study of the Transient Behavior and Stability Characteristics of Serial-Type Digital Data Systems," Communication and Electronics, (Trans.AIEE) pp 210-217, June 1959.

4. Bekey, G.A. "Generalized Integration on Analog Computers," IRE Transactions on Electronic Computers, Vol. EC-8, pp 210-217, June 1959.

5. Shumate, M. S. "Simulation of Sampled Data Systems Using Analog to Digital Converters," Proceedings WJCC, San Francisco, California, 1959.

6. Rawdin, E. "Time Multiplexing as Applied to Analog Computation," IRE Transactions on Electronic Computers, Vol. EC-8, pp 42-47, March 1959.

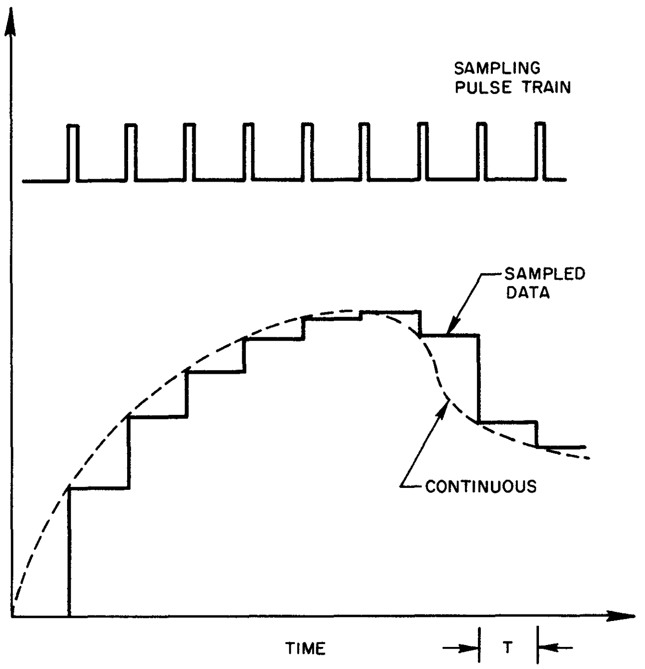


Figure 1.

Sampled-Data Input-Output Signal with Sampling Pulses.

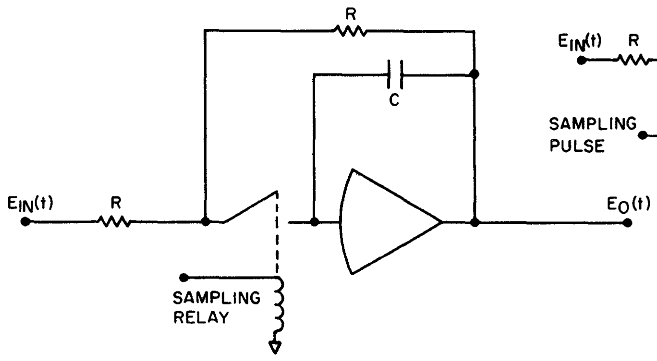


Figure 2. Sample-Hold Circuit.

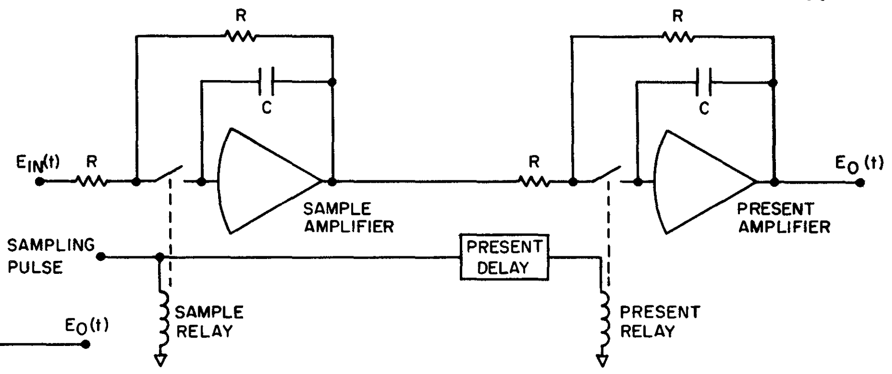


Figure 3. Sampled-Data Channel.

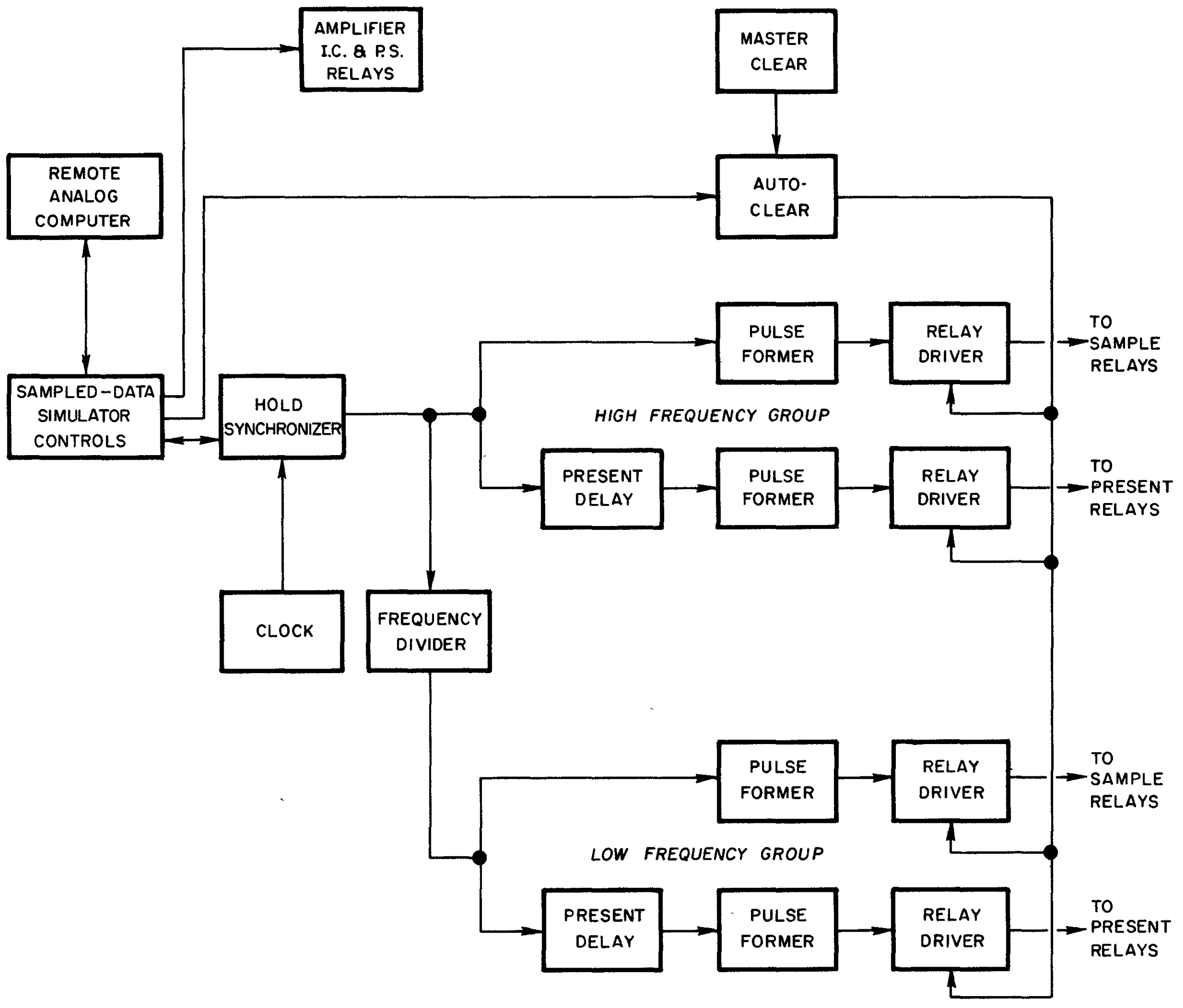


Figure 4. Basic Sampled-Data Control System.

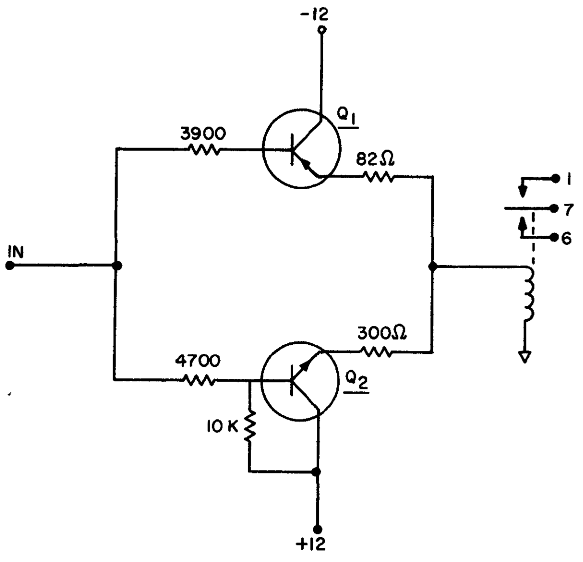


Figure 5. Relay Driver Circuit.

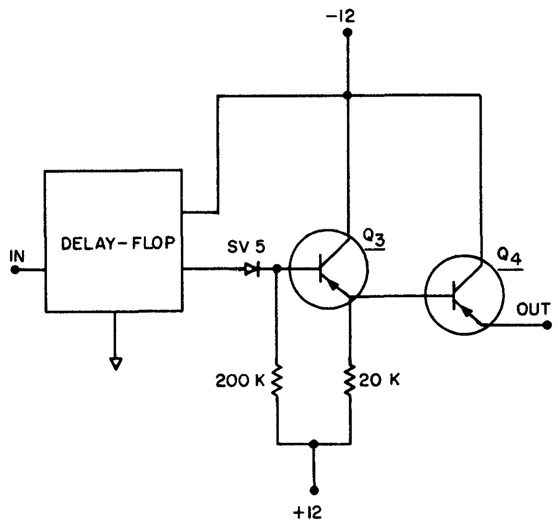


Figure 6. Pulse Former Circuit.

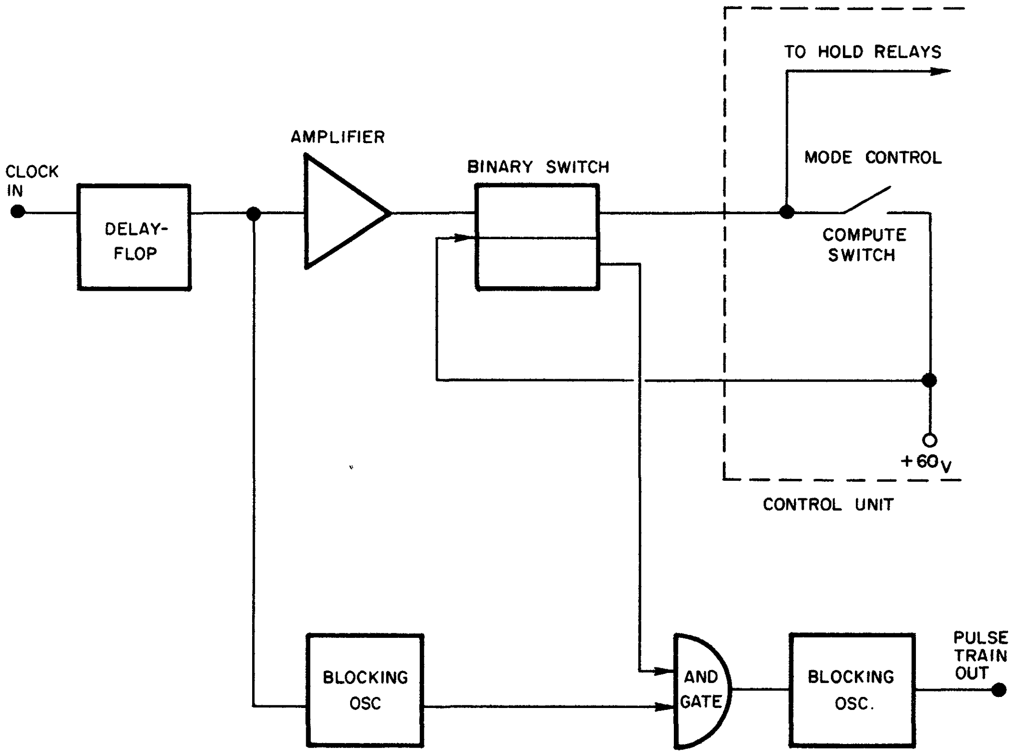


Figure 7. Block Diagram of Hold Synchronizer.

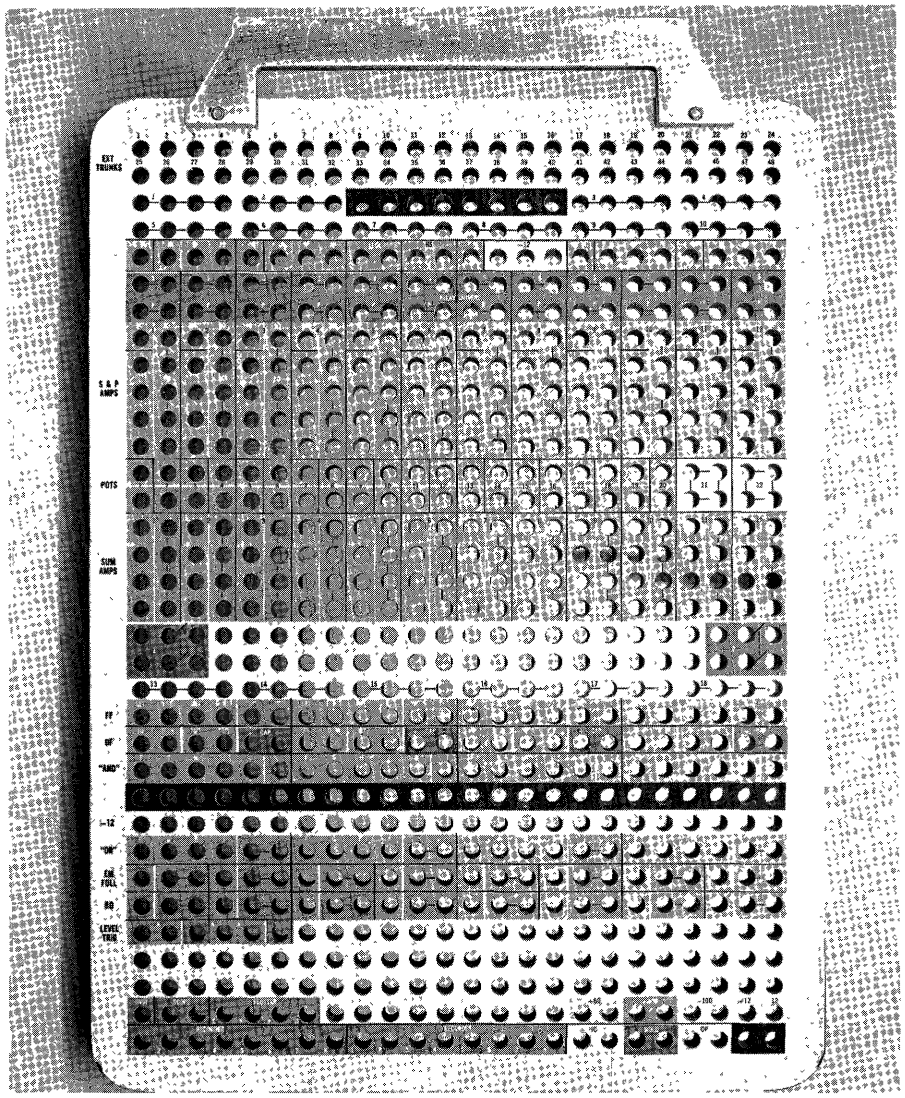


Figure 8. Patchboard.

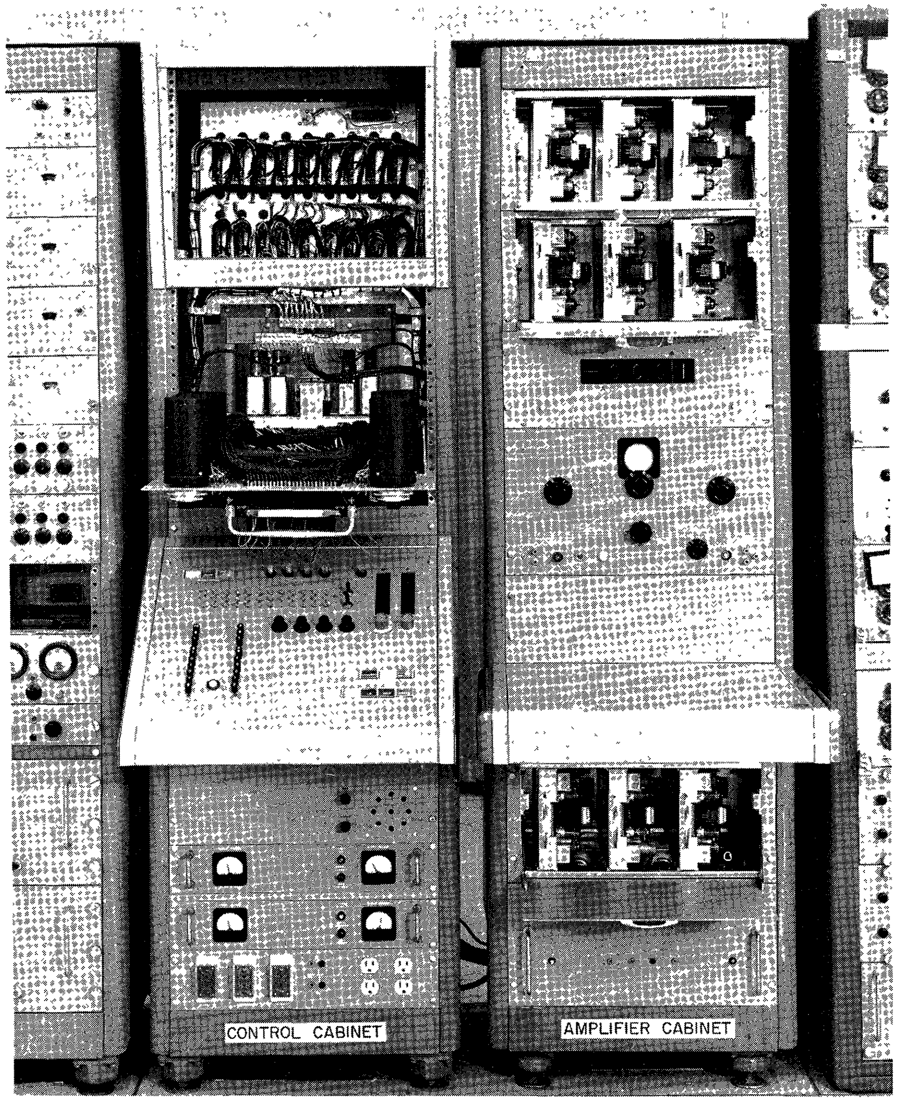


Figure 9. Simulator Control and Amplifier Cabinets.

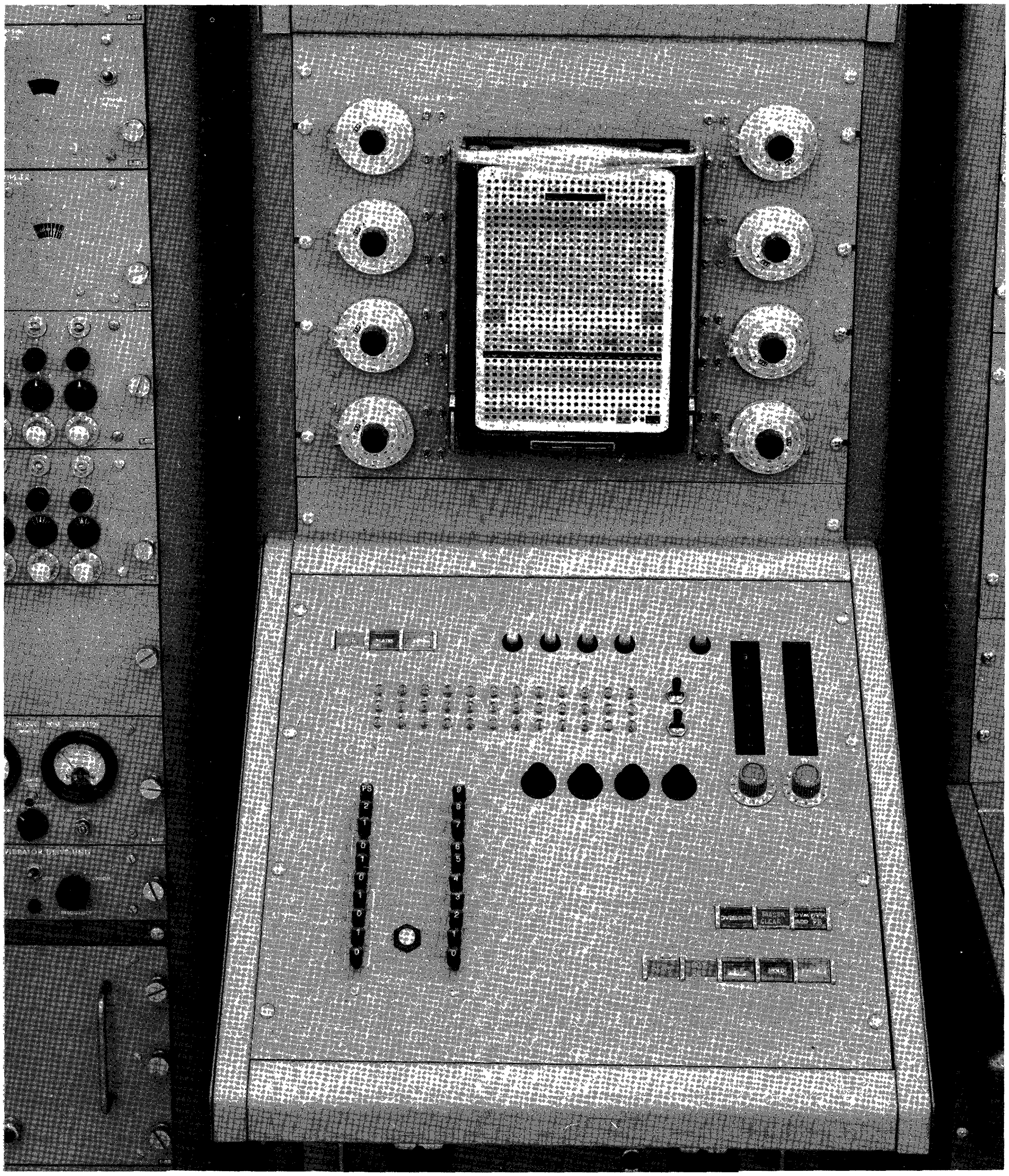


Figure 10. Control and Patchboard Panels.

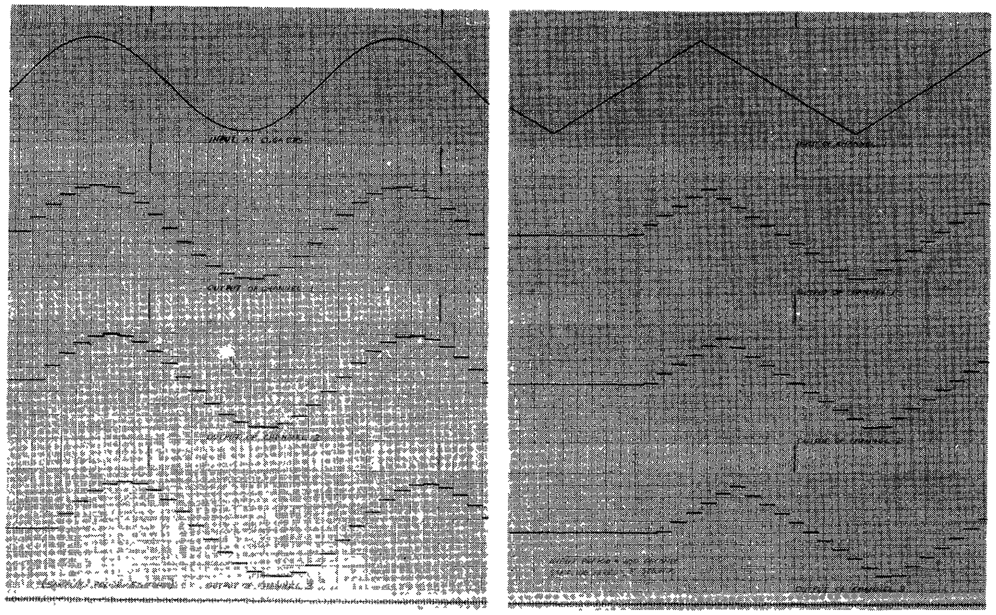


Figure 11. Input and Outputs of Three Cascaded Channels.

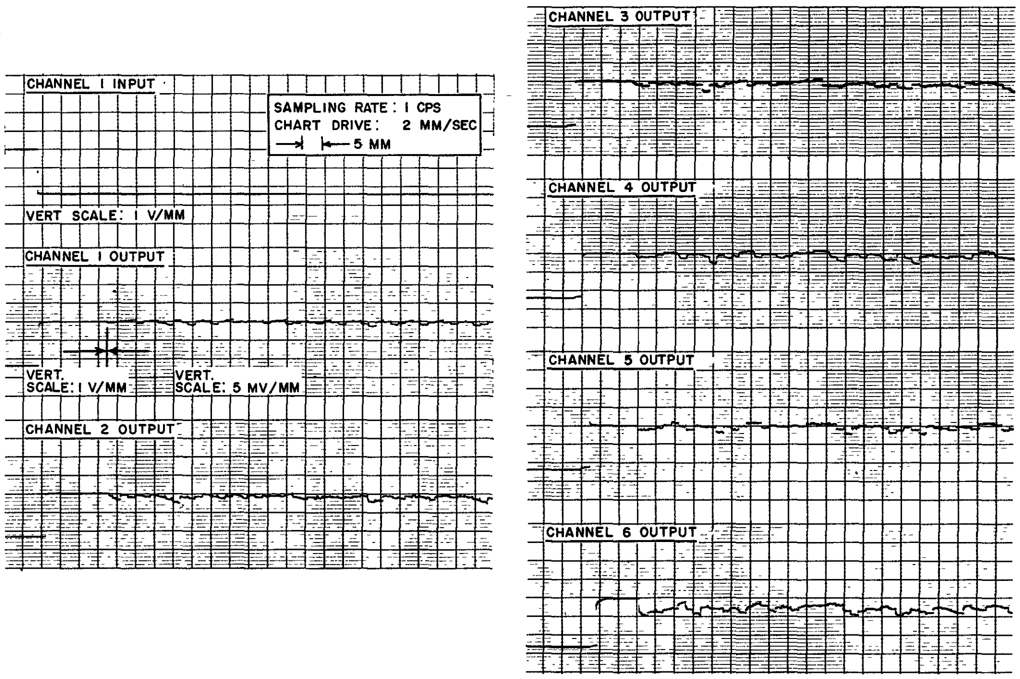


Figure 12. Comparison of Input and Outputs for Six Cascaded Channels.

A DIGITAL CONTROL UNIT FOR A REPETITIVE ANALOG COMPUTER

By Thomas A. Brubaker
and
Harry R. Eckes
Analog Computer Laboratory
University of Arizona
Tucson, Arizona

Summary

The University of Arizona's repetitive-computer control unit combines a 10 kc crystal oscillator with inexpensive preset decimal counters and simple digital circuitry to generate accurate timing pulses which perform the following functions:

1. Reset a repetitive analog computer at 100 cps, 50 cps, 25 cps, 10 cps, or on external triggering.

2. Actuate external equipment (statistical averaging computer) during a preset number of 1000 to 10,000 successive computer runs.

3. Furnish sampling pulses to sampling readout devices at push-button selected sampling times t_1 and t_2 (or $t_1 + \tau$) seconds after the start of each individual computer run.

4. Furnish variable-brightness oscilloscope timing markers at 1000 cps, 500 cps, 250 cps, and 100 cps. In addition, markers are available at the computer repetition rate and at times t_1 and t_2.

For statistical experiments, the 10 kc clock oscillator will be detuned slightly, so as to produce sampling rates not harmonically related to the 60 cps line frequency.

Principles of Operation

Figure 1 shows a repetitive-analog-computer set up in block-diagram form, together with a pictorial representation of four computer runs with random initial conditions, random parameters, and/or random forcing functions.

Referring to Fig. 1, the control unit furnishes accurately timed reset pulses to the repetitive-analog-computer. While the reset pulse is positive, all computer integrator output voltages are reset to their preset or random initial values (RESET condition of the repetitive-computer).[1] A typical computer run begins when the reset pulse is turned off (COMPUTE condition). At push-button selected sampling times t_1 and t_2 (or $t_1 + \tau$) seconds after the start of each computer run, timing pulses from the control unit cause two sample-hold circuits to store two selected computer voltages, $X(t)$ and $Y(t)$, so that sample values $X(t_1)$ and $Y(t_2)$ are read out. Separate pulses from the control unit terminate the computer run and reset the two sample-hold circuits so that they can track a set portion of the next computer run.

The computer cycle is typically repeated at repetition rates of 10, 25, 50, or 100 cps, with reset periods taking 10 percent of the repetition period. Thus, a 90 msec computer run would be followed by a 10 msec reset period. Computer runs and/or sample-hold circuits may also be triggered by external devices, such as analog or digital computers or external measuring equipment.

$$\text{Samples } \left[{}^1X(t_1),\ {}^2X(t_1),\ ..\ {}^nX(t_1)\right] \text{ or } \left[{}^1X(t_1)\ {}^1Y(t_2),\ {}^2X(t_1)\ {}^2Y(t_2),\ ...\ {}^nX(t_1)\ {}^nY(t_2)\right]$$

from a predetermined number of successive computer runs are used in the statistics computer to produce estimates of ensemble statistics, such as probabilities, probability densities, expected values, mean-square delay error and correlation functions. Sample sizes between 1000 and 10,000 are push-button selected on the control panel.

Block Diagram

Figure 2 is an overall block diagram of the digital control unit.

To begin a series of computer runs, it is necessary to reset all counters, binaries and bistable multivibrators to their proper states. If the manual reset button is used, all sample-hold circuits are reset to track, the analog-computer is in RESET and all counters read zero. For external resetting, pulses generated in the correct sequence are used to reset the proper counters and multivibrators for the desired operation.

Internal Clock Mode of Operation

Generation of Reset Pulses

Referring to Fig. 2, when operating in the INTERNAL CLOCK mode, the 10 kc crystal oscillator provides the input to a group of three preset decimal counting units, (preset DCU's 1, 2, and 3), which are used as frequency scalers. The output of each DCU is a positive square pulse, whose negative going edge may be differentiated to yield pulses at one-tenth the input pulse rate. To get sharp negative pulse trains at 1000 and 100 cps, the outputs of counters 1 and 2 are put through simple differentiating buffer amplifiers. Binary scaling of the 1000 cps pulses yields four negative pulse trains at 1000, 500, 250, and 100 cps, which are selected by the repetition-rate selector switch to give 10 times the desired repetition rate of 100, 50, 25 or 10 cps. These pulses are also used as oscilloscope timing-markers.

For a particular repetition rate, the selected pulse train drives a permanently "preset-nine" DCU (DCU7) The output of this scaler is positive between each 9th and 10th input pulse (Fig. 5b). This square pulse, whose length is one-tenth the computer run time, is the reset pulse used to restore the analog-computer elements to their initial conditions. The negative-going trailing edge of this reset pulse precisely marks the start of a computer run and differentiation yields the negative COMPUTE pulses, which can be counted to determine the total number of runs. These COMPUTE pulses are also used to reset various counters and bistable multivibrators and as time-markers.

Generation of Sampling Pulses

t_1 Pulses. To obtain a pulse t_1 seconds after the start of a computer run, decimal counters 1, 2, and 3 can be push-button preset to provide a pulse from 0 to 100 msec after the start of a run in 10^{-4} second intervals. At time t_1, the preset outputs of the three counters all go positive and the AND gate (A1) emits a positive pulse which is differentiated and inverted. This negative pulse turns on a bistable multivibrator (M1) which blocks any more preset counter pulses until it is reset by a COMPUTE pulse. The differentiated multivibrator output is the desired negative t_1 pulse which performs the following functions:

1. It actuates a bistable multivibrator (M4) which puts the t_1 sample-hold circuit in HOLD.
2. It serves as a t_1 time-marker.
3. It resets a permanently "preset-nine" scaler (DCU 8).

The input to this DCU is the pulse train at the frequency equal to 10 times the repetition rate. The output of the scaler is a square pulse (Fig. 5c) whose positive-going leading edge is differentiated and inverted to reset the t_1 sample-hold circuit into its tracking mode. The resulting sample-hold tracking time can vary from one-tenth to two-tenths the computer run time. If it were desirable to make this tracking time equal to exactly one-tenth the length of a computer run for all possible t_1, additional counters would be required; since the tracking time is not critical, one counter suffices.

t_2 Pulses. To obtain a pulse t_2 seconds after the start of a computer run, the $t_2 - \tau$ selector switch is placed in the t_2 position. Now the bistable multivibrator (M6) holds the AND gate (A5) on and DCU's 4, 5, and 6 are synchronized with counters 1, 2, and 3 to produce a positive output at the preset time t_2. If a sampling pulse is desired τ seconds after the first sampling time t_1, the selector switch is placed in the τ position. In this mode, the t_1 pulse resets counters 4, 5, and 6 and turns on the bistable multivibrator (M6) which, in turn, permits the AND gate (A5) to pass the counter input pulses from the crystal oscillator, so that scaling begins at time t_1. In either case, the pulse at the time t_2 or $t_1 + \tau$ is used to trigger a bistable multivibrator (M2), which in turn provides the negative pulse which actuates the t_2 sample-hold circuit, acts as a time marker and resets a "preset-nine" scaler (DCU 9).

The Total-Run Counter

The total-run counter counts each COMPUTE pulse as the end of a computer run, so as to count only completed runs. The run counter consists of two regular DCU's (10 and 11), and two preset DCU's (12 and 13), so that any integral number of hundreds of runs between 1000 and 10,000 can be push-button selected. When the selected total is reached, the preset outputs of DCU's 12 and 13 go positive, and the AND gate (A3) emits a positive pulse. This pulse is differentiated and inverted to trigger a bistable multivibrator (M5), which then turns off the AND gate (A4). This pulse also turns off all external statistical computing equipment, which now contains statistics of the system under study.

External Computer Control

In the EXTERNAL CLOCK mode, an external device supplies reset pulses, which must be negative during the COMPUTE portion of the run and must go positive to reset the analog-computer elements to their initial conditions. Here again, the trailing edge of the reset pulse is differentiated to provide a negative COMPUTE pulse for run counting, for resetting counters and multivibrators, and as a time-marker.

This COMPUTE pulse resets DCU's 1, 2, 3, 4, 5, and 6 which are preset to give pulses at times t_1 and t_2. The only difference is that now the "preset-nine" DCU's 8 and 9 operate at a 100 cps input rate, which means that the computer repetition rate should not exceed 10 cps. This maximum repetition rate can be easily varied if faster rates are desired. If external t_1 and t_2 pulses are used to actuate the sample-hold bistable multivibrators (M3 and M4), sample-hold reset pulses must also be supplied to reset the sample-hold circuits into track between successive samples.

Oscillator, Buffers, AND Gates, and Bistable Multivibrators

Figure 3 shows the 10 Kc crystal oscillator and inverter circuit. The circuit operates so as to produce a non-sinusoidal output waveform of amplitude 90 volts which has a sharp fall time of about 3 μsec.

Figure 4 shows the following circuits:

a. Diode AND gate and inverter. This gate uses the preset counter outputs as the diode inputs to give a positive pulse when all inputs are positive. The pulse is differentiated and drives an inverter biased to cutoff, so that the final output is a sharp negative pulse.

b. Cathode follower AND gate. Here a bistable multivibrator output controls the bias on a cathode follower. If the multivibrator output is positive, the tube conducts and the differentiated input signal is passed.

c. Buffer amplifier. This circuit differentiates the input pulses and clips off their positive going portion. The results are sharp negative pulses from a low-impedance source. For use as a dc coupled cathode follower, the input network is removed and the circuit is dc coupled through a voltage divider to the input to provide a square pulse output at the proper voltage levels.

d. Bistable multivibrator. When the input is taken through the diode network the circuit acts as a binary scaler. For use as a bistable device for driving gates the diode network is removed and the inputs are the 50 pfd capacitors.

The digital control unit operates from the $\pm$ 300 volt repetitive-analog-computer power supplies. For reasons of economy vacuum-tube components were used, however, the system would lend itself to implementation with standard transistor modules.

Acknowledgements

The circuit was developed in the course of a repetitive analog computer project directed by G. A. Korn. Acknowledgement is due the Electrical Engineering Department of the University of Arizona and Dr. P. E. Russell for continuing support of this work.

References

1. Huskey, H., and G. A. Korn, Computer Handbook, McGraw-Hill, N.Y., 1961 (in print).

2. Millman, J. and H. Taub, Pulse and Digital Circuits, McGraw-Hill, N.Y., 1956.

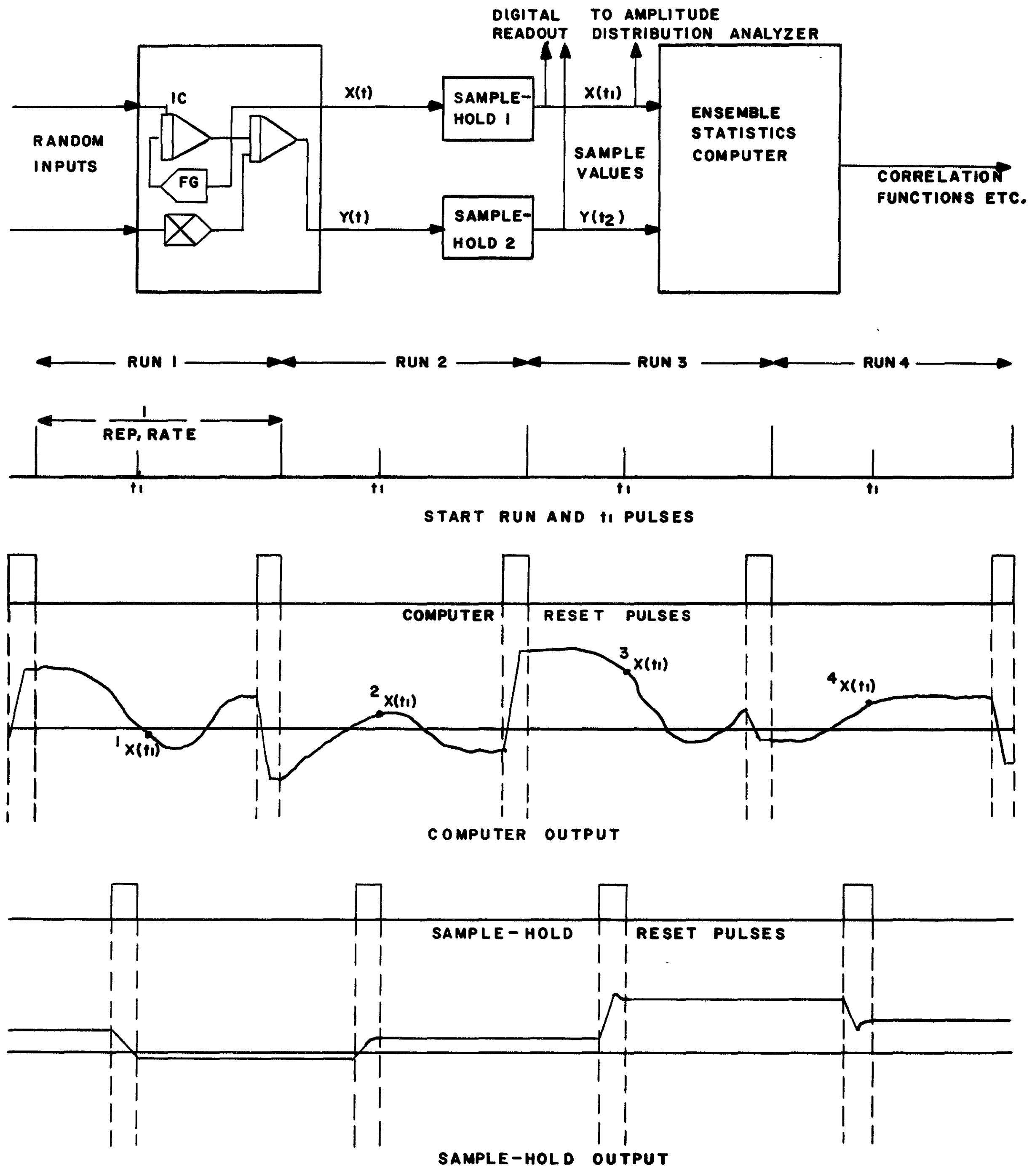


Fig. 1

A Typical Repetitive-Analog-Computer Set Up Together With Four Sample Computer Runs

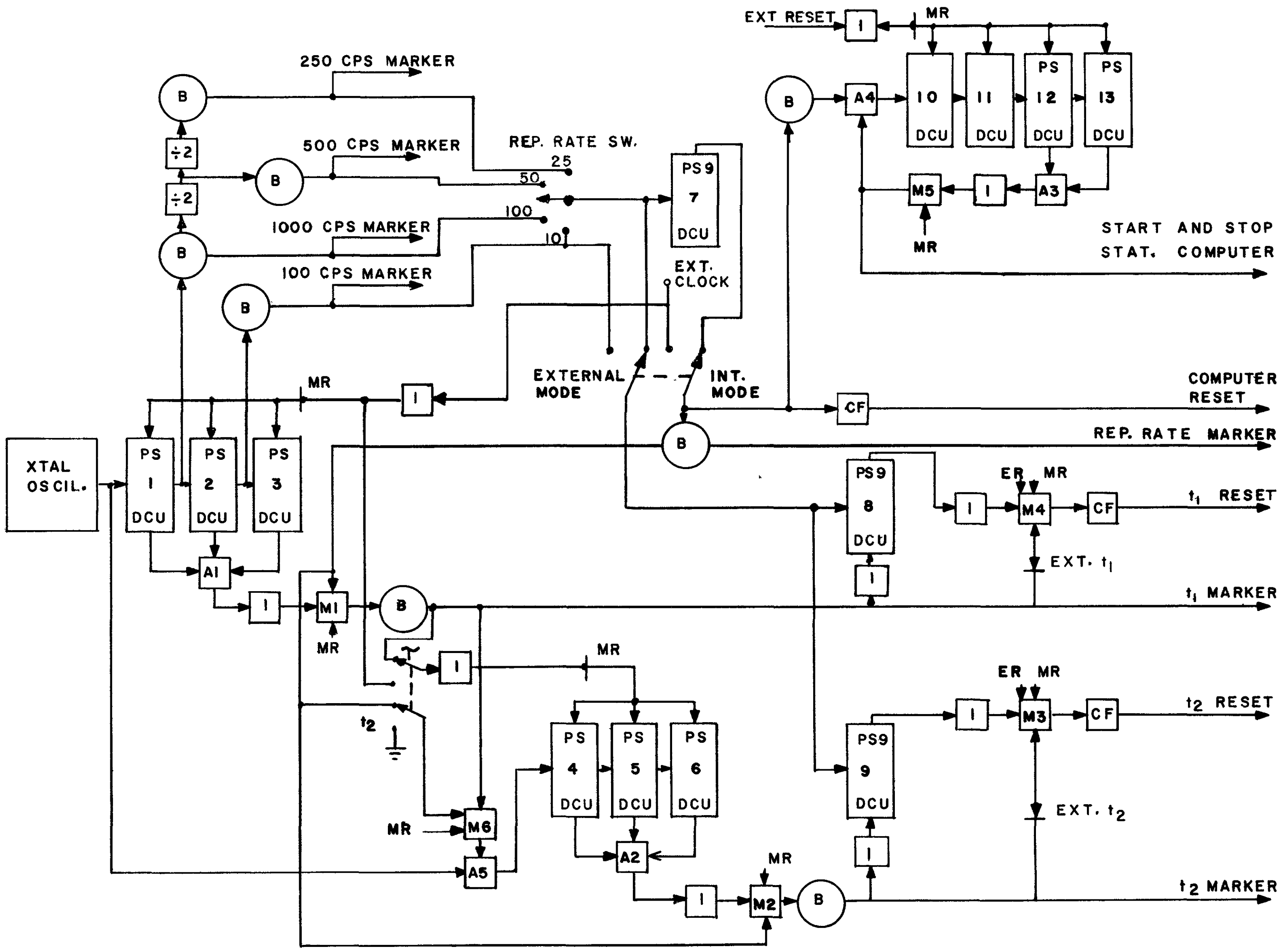


Fig. 2

Digital Control Unit Block Diagram

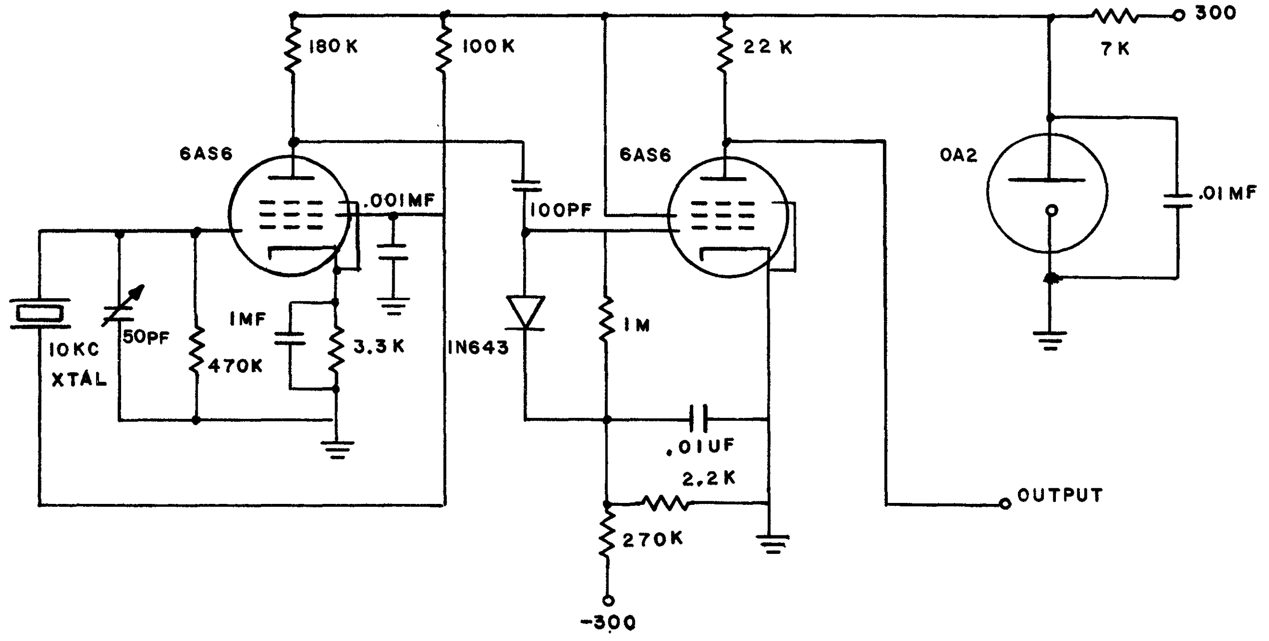
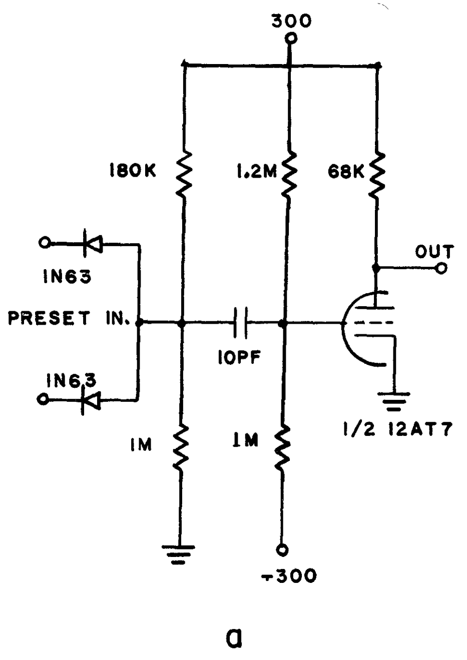
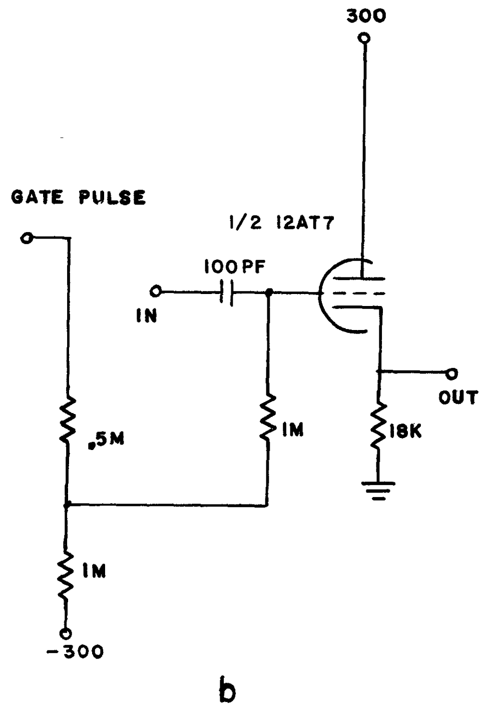
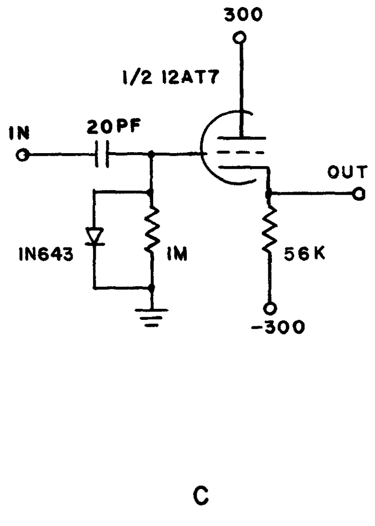


Fig. 3

10 KC Crystal Oscillator







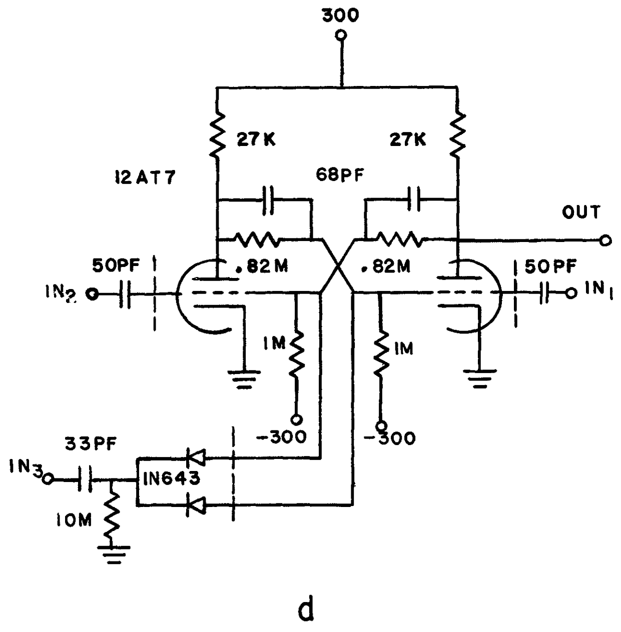


Fig. 4

a. Diode AND Gate and Inverter

b. Cathode Follower AND Gate

c. Buffer

d. Basic Bistable Multivibrator

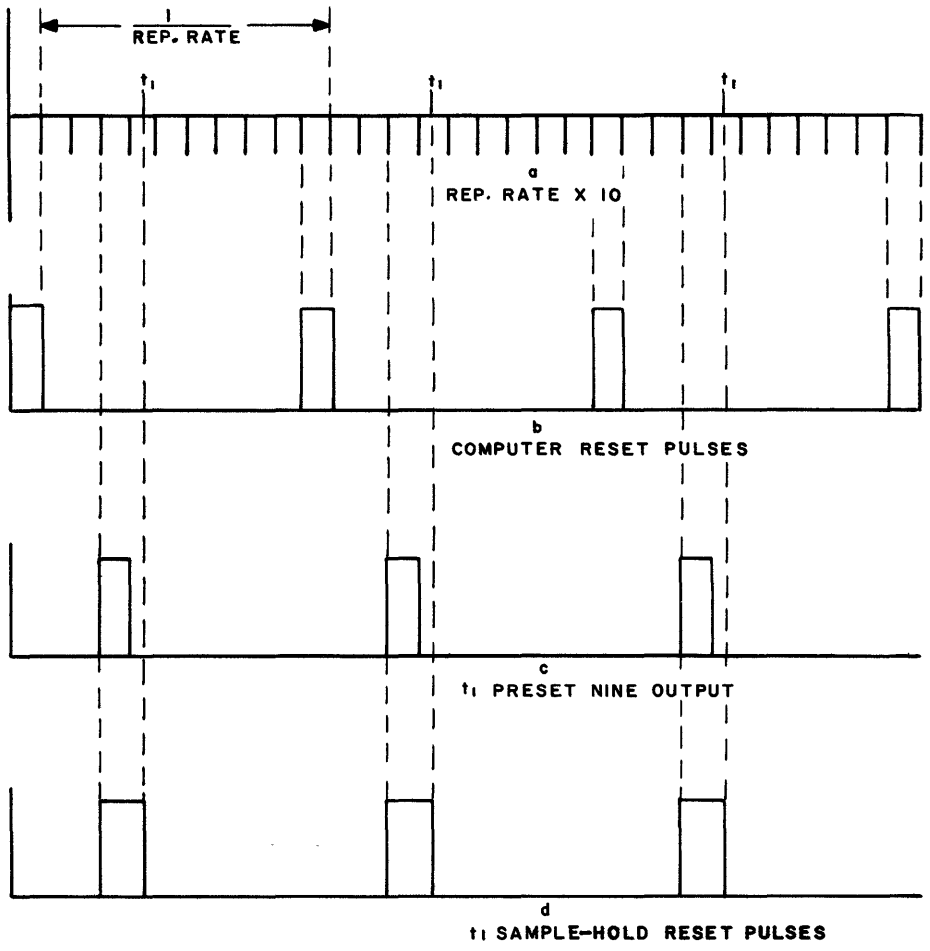


Fig. 5

Reset Pulse Diagram

TRENDS IN DESIGN OF LARGE COMPUTER SYSTEMS

Charles W. Adams
Charles W. Adams Associates, Inc.
Bedford, Massachusetts

Summary

New developments in computer design are reported and trends are analyzed -- first with regard to physical devices with emphasis on fixed and high-speed storage systems; then with regard to logical techniques including various logical organization schemes, stored logic, concurrent autonomous operation, and new approaches to modularity in design.

Introduction

When the WJCC Program Committee invited me to prepare a survey of newer, larger computer systems, they suggested that I include both an indication of what are the characteristics of the specific systems and an interpretation of what these imply for the design of future systems. It was also their intention that computer manufacturers be encouraged to give five-minute discussions of their new systems.

I have taken the liberty of changing that format slightly, by considering different aspects of computer design and using specific systems as illustrations, and have sought to find "interlude" speakers who would give broad, unbiased, expert coverage to important design aspects.

In preparation for the session, a rough but fairly comprehensive outline was prepared and distributed to a dozen or so friends who are well-known both for their awareness of new developments and trends in the computer field and for their willingness to express themselves freely. Based on their thoughtful replies, as well as on material received from a number of people representing the computer manufacturers who had new systems and devices to report, I came to an almost obvious conclusion: the keynote in the design of new systems, whether large or small, is not hardware but logic. Consequently, new developments in physical hardware can be rather quickly summarized, with principal emphasis given to new approaches to internal logical operation.

New Devices

The hardware development which caused so much excitement only a few years ago -- solid-state circuitry, two-microsecond core storage cycles, magnetic tapes operating reliably at upwards of a hundred thousand characters per second, magnetic juke-boxes for "random" access to files of ten or twenty million characters, scanners for reading and interpreting printed numbers optically -- are now taken for granted. Today the attention of the system designer is turned toward facilitating the effective utilization of these now almost humdrum achievements in hardware design.

This is not to say that new developments are not being made in the hardware area. For one thing, rather spectacular improvements are being made in the reliability and costs associated with all aspects of computer componentry. Most notable perhaps are the continually improving performance of the large drums, disc files, optical scanners, and magnetic-ink character recognition devices (witness the new checks the banks are supplying to all their customers as part of the nation-wide adoption of MICR).

A new approach to random-access files has been announced by NCR, to wit the CRAM system involving automatically selected magnetizable sheets which are then wrapped around a drum. IBM demonstrated last year a "tractor" system for selecting tape reels automatically. The really new hardware seems, however, to be appearing primarily in the area of high-speed storage, both erasable and non-erasable.

Non-erasable Storage

Increased attention is being given to various forms of non-erasable (i.e., read-only) storage systems, large and small. Though new in detail, such devices are far from new in principle. Interest in them stems in part from increasing recognition of the potentialities of using "stored logic" in system design, and in part from the growing popularity of special-purpose stored program and/or stored data file systems.

One example of non-erasable storage is the 0.2 microsecond magnetic slug memory which will be used in the Ferranti Atlas computer to contain an executive routine for multi-programming control as well as such things as macro-program subroutines. Speed is the most important aspect of this device, but the fact that it can be altered only at the factory puts the Ferranti programmers in an enviable position indeed since no user can tamper with their programming package once it leaves the plant.

Various control computers use non-erasable memories such as drums with mechanically locked-out recording circuitry, twisters biased with permanent magnets, and non-destructive reading from thin-film storage. For example, Remington Rand announced some time ago a dual-film device in which an erasable cobalt film biases a read-only permalloy film, and is actually using Bell Labs twisters with magnets mounted on removable cards. In the M490 and 1206 they provide a small diode matrix, the contents of which can be mechanically altered only by inserting plugs pre-assembled with wires soldered in place, for use in loading error routines, etc. And, of course, more than ten years ago M.I.T.'s Whirlwind I was performing 125,000 instructions per second, working from a storage of 80 flip-flops and 432 diodes controlled by individual toggle switches.

A third class of non-erasable storage is photographic storage, optically scanned by cathode ray tubes, which yield a very large high bit-rate file -- e.g., the IBM photoscopic disc memory for use in language translation and the BTL random-access photographic storage used in electronic telephone exchanges. In these cases, the emphasis is large volume data storage at reasonable costs and/or rather high speeds, while the fixed drums, twisters and diodes are aimed at protection against accidental loss of information (usually of a real-time control program).

High-speed Storage Devices

Reading or writing in a random-access storage in less than a microsecond is being accomplished in thin-film magnetic memories on which the M.I.T. Lincoln Laboratory, Remington Rand, and Honeywell have announced some results. Small (128 word in the UNIVAC 1107 case) thin film storages operating at 0.6 microseconds are now functioning satisfactorily in the laboratory. A ten-thousand-word storage at 0.1 microseconds is the present objective at Lincoln Laboratory.

The National Cash Register Company has announced a magnetic rod memory storing a thousand bits per cubic inch with switching times of 0.05 microseconds.

Aside from film and rod systems, work is apparently still continuing in the cryogenic area (involving superconductivity phenomena at temperatures near absolute zero), but no real breakthroughs have been made public.

Increasing Effective Storage Speed

Techniques for increasing the effective speed of a conventional magnetic core storage device include using two or more independent banks alternately, providing asynchronous operation in which rewriting is delayed if possible until storage access is not otherwise needed, and implementing look-ahead schemes by which any potentially idle time is used to read information in anticipation of its later use. The gain from such techniques is limited to a reasonably small percentage increase (under 100%) and certainly in practice has not always been as great as has been expected. In any event, these are basically logical rather than physical means of increasing speed, akin to the various forms of autonomous operation discussed later.

New Logical Designs

The biggest design improvements in large digital computers during the next few years seems likely to come in the logical organization of the systems rather than in the componentry involved. The techniques employed and the objectives gained are many and various --- so much so that what follows cannot lay claim to being even a comprehensive catalog, quite aside from not containing any appreciable degree of detail. What has been attempted, rather, is to make mention of a number of techniques and to assess their apparent purpose and merit.

Arithmetic

The age-old problem of choosing among binary, decimal, and alphanumerical operation has not been completely resolved, but the natural tendency is toward compromise: binary computers which have convenient facilities for conversion to decimal and even machines with two or more forms of operation built directly into the hardware (e.g., the Honeywell 800 and the RCA 601).

Effective storage capacity is sometimes increased by use of a short word length with built-in double-precision arithmetic to be used when needed without an excessive speed penalty (e.g., the Ramo-Wooldridge AN/UYK-1 and the Packard Bell 250). Alternatively, provision is made for dealing with half-words as in the Remington Rand 1107 and the IBM 7030.

Most large machines of course have built-in floating point arithmetic at least as an optional feature, but the Bendix G20 is the only one being promoted as having no fixed point operations. (Provision is made for unnormalized floating operations which then are essentially fixed point). However, one sometimes encounters business men who will not consider the G20 because they definitely want to be able to do fixed point arithmetic.

Address Logic

Merely numbering storage locations consecutively from 0 to N is now old hat. Indirect addressing and "literals" permit instructions to operate on the contents of the location whose address is contained in "x" or, much more directly, on "x" itself. Both are convenient in certain cases, but designers who haven't bits to burn in their instruction words sometimes omit these features.

On the other end of the scale, when there are more storage registers than there are bits to distinguish between them in the instructions, bank addressing (Honeywell 800) or relative addressing (CDC 160) are sometimes used. The programmer often has little patience with these --- to him they are merely a nuisance. But the obvious economy of storage they permit (since most instructions do, or can be made to, refer to nearby locations) has perhaps not been as widely recognized as it should.

Engineering problems occasionally deal with short numbers, but usually with fairly long ones, and never with very, very long ones. The business user, however, concerns himself with "fields," not variables, and his fields can and do run the gamut from a single bit (male or female?) to several hundred bits (home address). Character-addressable machines like the IBM 705 and RCA 501 (and many others) help dispose of this problem, but a more honest way of dealing with the situation appears to be "field addressing," as in the IBM 7070, in which the existence of words is admitted (the IBM 705 has of course a 5-character word length, but evidently is ashamed to mention it). Semantics aside, character addressability would be a very desirable feature if it could be accomplished in a parallel manner to preserve speed.

A novel addressing scheme is planned for the Ferranti Atlas computer. It is a "one-level store" in which a large drum logically comprises the main memory, each drum location being separately addressed, yet the programs actually operate from a large magnetic-core working storage. "Pages" of 512 words are "automatically" brought into core as needed, and copied back to the drum when the space is more urgently needed for another page. Every reference to storage requires that the page number be first processed against a list of all the pages already in core, and the appropriate core page used where possible. An executive routine in the "fixed store" (mentioned earlier) is used to decide which page to replace if the reference is to a page not already in core. The scanning of the list is done in a parallel fashion in a fraction of a microsecond.

Instruction Format and Repertoire

Of the 35 commercially announced solid-state general purpose computers, 24 are single address and only the Honeywell 800 and 400 and the NCR 304 are three-address. The IBM 1401, 1410, and 1620, all the RCA systems, and the RW400 are basically two-address machines but in a few cases four, three, two, one or no addresses are used in different operations.

More startling yet is the logic of the Burroughs B5000 which, they assert, "can well be considered the first non-vonNeumann computer." Operations are grouped into classes and the computer operates in arithmetic mode, subroutine mode, data-manipulation mode, or control mode, the last involving an executive routine permanently recorded on a magnetic drum. In arithmetic mode, the computer interprets a string of intermingled addresses and operations written in "Polish notation" (the expression $y = (w + i + t)\ (p - q)/z$ becomes $ywi + t + pq - \cdot z/ =$ in Polish notation).

A number of interesting operation codes appear in some of the new computers, with the Remington Rand 1107 certainly among the leaders in the aspect of design. It is the stored-logic approach that holds the greatest fascination, however. The Ferranti Atlas, for example, uses one bit to make operation codes which are used as any built-in code but is executed by the computer as a subroutine stored in the fixed store. The Ramo-Wooldridge AN/UYK-1 on the other hand has no elaborate built-in operation at all and uses "lograms" of simple instructions to carry out more complex operations. Means are provided to go from logram to logram automatically, so that in use they behave much the same as conventional built-in instructions.

Autonomous Operation

To make effective use of each of the expensive high-performance magnetic tape units, core storage banks and arithmetic-logical elements in a large computer, each must be kept as busy as possible. Since different problems make different demands on the various units, proper balance can only be obtained by operating several programs at one time in the hope that the combination will provide a better distribution of demand. If a balanced load is to be thus achieved, each unit must be designed to operate autonomously, under its own control, and there must be an executive or scheduling procedure that keeps everything running as smoothly as possible.

The approaches to the control of a large system of autonomous units are several. In the IBM 7090, Bendix G20, and others, for example, elaborate input-output control units work out of the same storage as the central control, and means are usually provided to "trap" or interrupt the main program when the input-output control needs a new set of instructions. All of the scheduling is done by a part of the regularly stored program, although most users do not have to concern themselves with the question of this "driver" or "executive" routine.

The Honeywell 800 uses a highly-publicized multi-programming arrangement, in which each of up to eight different programs are performed virtually a step at a time in sequence. While a program is waiting for an input-output unit to function, it is by-passed so that there is no idle computer time.

The Ferranti Atlas fixed store is used to hold an executive routine which is automatically called in whenever any autonomous unit needs attention and whenever the central computer is faced with any delay for input-output in the program currently being processed. Thus, the executive routine carries on moment-by-moment control of the scheduling, intermixing programs as required to give the best possible utilization of the system.

The Atlas also provides multiple consoles so that the separate programs may have separate operators. Furthermore, the executive routine can alter the page reference list (see above) according to what program is in use, so that the same "drum" location can be referred to by more than one program yet have no interference between them (the executive routine keeps track of the actual drum pages corresponding to the page numbers used in each program --- the absolute addresses are in effect treated symbolically during operation).

The multiple consoles of the Atlas or the Bendix G20 make possible concurrent "on-line" program debugging by several programmers at one time. The Atlas scheme provides foolproof protection of each program against accidental alteration by another sharing the machine at the same time. The potentialities and techniques of concurrent (autonomous) debugging would be a worthy subject for considerably more discussion than can be provided here.

Modularity, Graceful Degradation

Intimately associated with the provision for and control of autonomous operations of control computers, storage banks, drums, tapes, in-out devices, consoles, etc., is the question of modularity and of providing for graceful degradation. This latter term is unfortunately not familiar to many computer designers -- as will be seen, it refers to the behavior of the system in the face of malfunctioning of one or more of its parts.

Most present-day large computers are modular in that the user has a wide selection of usable configurations and correspondingly of system rental costs. In most cases, failure of a peripheral device affects only that part of the system, and the rest of the computer can continue in operation. But when any part of the central computer fails, the system is usually completely out of action.

Graceful degradation is a design objective and for large systems is an important one. It implies that multiple central control units and storage banks are provided and are treated as autonomous units along with the tapes, etc. Using the philosophy of a telephone exchange, components are assigned to jobs as needed, and malfunctioning units are by-passed.

Conclusion

It seems safe to predict that computer hardware will continue to gain in speed and particularly in reliability as the years go on. Almost equally certain is a continued trend toward stored logic and toward more autonomous operation in a multiprogrammed modular computer capable of some degree of graceful degradation. The results in terms of improved performance per dollar should be impressive.

CURRENT PROBLEMS IN AUTOMATIC PROGRAMMING

Ascher Opler

Computer Usage Company, Inc.

18 East 41st Street

New York 17, New York

Summary

The proliferation of the number of applications to be programmed and the number of types of computers available has created a significant and challenging problem. This survey will consider some of the more promising suggestions for enabling automatic programming to keep pace with other developments in the field. Among the prospects are: standardization of source languages, development of a standard universal intermediate language, design of computers to operate directly in source language, automatic translation of programs between computers, the automatic construction of compilers and the standardization and construction of common compiler modules.

Automatic programming has grown from an interesting child to a troublesome adolescent in the past few years. The number of publications and meetings devoted to its problems has continued to increase. This presentation will attempt to state the problems from a practical viewpoint and to survey a number of techniques that have been proposed to handle some of the current difficulties. The viewpoint taken here is a practical one and considerations of costs, staff, delivery time, competitive position, etc., will not be foreign to the discussion.

The number of groups developing automatic programming (and closely related systems) today is enormous and continually increasing. Personnel so involved are primarily (a) representatives of the computer manufacturer, (b) system programmers at the more advanced computer installations and (c) those involved in computer programming research (most frequently at universities and large independent research organizations).

With time, the responsibility for developing system programs has shifted from the user to cooperative groups and now to the manufacturer. Computer manufacturers are expected by their customers to supply with the machine a complete set of automatic programming systems. With the field becoming highly competitive, each manufacturer is expected to deliver the automatic programming package at the same time the computer is delivered. Manufacturer's representatives are asked questions concerning the programming package and its availability more often than they are asked questions regarding the hardware and its availability.

Dependence upon the applied programming staff of a manufacturer is not completely universal. A dozen or so highly experienced computing equipment users have developed their own systems either because they believe them to be superior to those of the manufacturer, more compatible with their own operating methods or can be made operational months before equivalent programming systems promised by the manufacturer. The achievements of this group has been particularly impressive.

Most automatic programming systems built by manufacturers have been based on fairly solid concepts developed over the past few years. Fortunately, there are a group of brilliant energetic researchers in the automatic programming area who have been making rapid strides toward developing greatly improved techniques. Some of these contributions will be reviewed in the light of the overwhelming problems confronting the programming field today.

Mounting Pressures on Developers of Automatic Programming Systems

Those who would develop automatic programming systems for production purposes are faced with many pressures. Some of the most troublesome are outlined below.

A. Increased Requirements

At the present time, the automatic programming systems supplied by a manufacturer (and expected by his customers), fall into two categories: (1) compilers and (2) other systems (including assemblers, operating and debugging systems, sort generators, etc.). For most of what follows, the discussion will center on the compilers (defined as programs which translate from a source language (usually problem oriented) to a machine language). In the area of automatic programming systems, the manufacturer is expected to supply compilers to translate from one or more standard algebraic languages and from one or more standard commercial languages. Furthermore, these compilers should be able to operate effectively on any of the numerous configurations of equipment that one may order and install and to produce object programs that are also effectively operable on any of this wide degree of equipment modularity. It is not uncommon for the purchaser of equipment to request assurance that the manufacturer will supply compilers to translate the same language for all successor machines that he may market in the future. A large manufacturer may very well have as many as ten compiler projects going simultaneously to prepare translators for several current, several announced but undelivered and perhaps one or two unannounced machines.

These requirements for "general purpose" languages for "general purpose" computers also carry over into the area of special purpose languages (automatic programming for numerically controlled machine tools; natural language translation; special military task programming, etc.) and into special purpose computers. At the present time one might hazard a guess that the amount of automatic programming effort in the category of the "special purpose" areas is approximately equal to that in the "general purpose" areas.

B. Shortage of System Programmers

The task of constructing automatic programming systems is generally relegated to the direction of those familiar with the art. Up to the present time, it has been a highly specialized skill and those who practice it are much in demand at premium salaries. The number of competent experienced system programmers is very few while the demand is very high. There is also at present very little concerted effort to train for this specialized programming area.

C. Mounting Costs of Automatic Programming

At the present time, the development of a compiler will range from less than \$100,000 to over \$1,000,000. These costs include the man months spent in programming and check-out, large quantities of machine time, great documentation efforts, education, training manuals, etc.

D. Developmental Time

The time to develop a compiler today will vary from six months to better than two years depending upon the type of compiler required, its quality and its associated features (diagnostic system, restarts, compatibility, modularity, etc.).

E. Simultaneous Development of Computer and Compiler

Because of the demand by customers that manufacturers deliver compilers starting with the first machine, an added pressure is placed upon the supplier. He must program and check out the compiler during the period of construction and testing of the computer prototype. Therefore, the compiler work is hampered by the necessity of using large amounts of simulation on another computer, working on an engineering model and operating without the availability of programming and debugging systems which are being concurrently developed. Considerable difficulty will also arise because of the large quantity of machine time required for effective check-out and testing of a compiler.

It is easy to see, under the extreme pressures created by the competitive marketing situation (which itself results from the rapid acceptance of automatic programming) and the continual spread of computer technology into new areas, that the pressures on developers of automatic programming systems are enormous. It is little wonder that they are rapidly scanning the horizon looking for developments that will reduce their costs and enable them to deliver new automatic programming systems rapidly and with a limited staff of system programmers.

Let us turn our attention now to some of the solutions that have been offered. These are divided into three areas: Elimination of the necessity of writing compilers, minimization of the number of compilers to be written, and minimization of the effort of writing each compiler.

Elimination of the Necessity of Writing Compilers

Recalling the well-known equivalence of hardware and programming leads one to consider the possibility of designing computers that will accept and directly execute programs written in source languages. Thus, in effect, the burden of translation would be eliminated and the source language would be the machine language.

From the fundamental nature of a Turing machine, one may deduce that, while the crude input could scarcely be executed directly, it is feasible to automatically convert the input to some directly useable form by programming. To pursue this further, consider a program available for the IBM 1620 that is called "GOTRAN". This program accepts FORTRAN statements and produces as output (in a single manipulation) the results of executing the statements. If one had this computer with the "GOTRAN" program locked-in, then he could consider it to be a machine that directly accepts and executes source language. However, this is really chimerical because the actual effort of writing "GOTRAN" was certainly of the order of magnitude of writing a compiler and thus we have actually eliminated no programming effort. What has been accomplished is the "conversion" of a general purpose computer to a special purpose device via programming.

For the achievement of the equivalent of programmed source language operation but with elimination of much of the programming, one would have to move closer to hardware developments. Two approaches that offer promise are the use of micro-programming and the design of computers with compiling requirements as a primary criterion.

The use of micro-programming has been known and discussed for some time. We hear much more today about the "customized" computer that can be micro-programmed to order. With a micro-programmed machine, it might well be possible to build aggregates of micro-steps that would carry out compiler instructions. Thus, one step toward the computer that would process source statements directly is the avoidance of construction of any logic higher than a micro-program step. The same question now appears - is the effort of micro-programming the machine to accept source language any less than that of constructing a compiler for a general purpose machine?

The other approach, to be described in the paper by R. S. Barton later in this session, lies closer to computer design. Ultimately, logical designers and automatic programming experts will be forced to work together.

Minimization of the Number of Compilers Required

A. Standardization of Source Languages

One effort that has been proceeding for the last few years and apparently producing considerable success is the cooperative movement to standardize source languages. There has been considerable international effort in the development of ALGOL and considerable national cooperation in standardizing on COBOL. Standardization will certainly continue in these areas. Ultimately, the fate of standardized languages will depend upon their acceptance. No matter how many agencies endorse a standard language, if the user finds a non-standard language compiler available which meets his needs, he will make no effort to use the standard language. Perhaps standard langauges cannot be accepted until non-standard languages are banned or plans are made to phase them out over a period of time. An interesting paper will be presented by Miss J. Sammet at this session suggesting even further reduction in the number of standard source languages.

B. Use of a Standard Intermediate Language

A proposal was made several years ago to interpose between the problem oriented languages and the computer oriented languages a universal computer oriented language (UNCOL). In a greatly oversimplified manner, if there are n problem oriented languages and m computer oriented languages, it requires (n x m) translators to translate from each POL to each COL. If one goes through the intermediate step and writes n translators from POL to UNCOL and m translators from UNCOL to each COL, then one replaces (n x m) translators with (n + m) translators. The present status of this development will be reported in a paper in this session by Mr. Thomas Steel.

C. Solutions to the Component Modularity Problem

With modern computers available as a collection of modules (of internal memories of various sizes, various input/output, buffering and control systems, magnetic drums, tape, discs, etc.), it becomes increasingly difficult to handle the two configuration problems, the compiling and the object complement of equipment. Thus far solutions offered to this general problem have not been common. Holt and Turanski have discussed an Allocation Interpreter for the object configuration problem. The general solution for the compiling configuration problem has been the simple change of table sizes and buffer sizes (thus speeding and expanding compilation) to adjust to the internal storage requirements during compilation. The answer to this problem when dealing with hierarchical memories with different access methods and speeds is sorely needed. The answer frequently offered by the manufacturer is to arbitrarily waive all but one or two compiling and object configurations. It is hoped that techniques will be forthcoming in the next few years that will allow each installation to make optimum use of whatever computing equipment is available to them.

D. Automatic Translation Between Source Languages

This is a possibility only in the special case where one source language is (or can be converted to) a subset of another. Fortunately, the FORTRAN language (which has become so common on many existing computers) is amenable to translation to ALGOL. At the present time, several FORTRAN-to-ALGOL translators are being written. This may also appear to be the relation between several data processing languages and COBOL.

E. Automatic Translation Between Object Languages

This is an area that is now being quite actively investigated. At one time, it was believed that the only method of source language to source language translation was by means of interpretive simulation. A considerable amount of research has been carried out recently in reexamining the possibility of machine language to machine language translation at a higher level and preliminary reports are encouraging.

Minimization of the Effort of Compiler Writing

A. Compilers Capable of Writing Other Compilers

This has been the subject of a tremendous effort in the past few years. When the idea of automatic programming was first proposed, one of the various suggestions was that, by "bootstrapping" with one operating compiler, one would be able to construct compilers for other machines, for other languages, and in fact even better compilers than the original one. While it is perhaps not universally agreed upon, this has been, in general, a disappointment. It has been repeatedly demonstrated that one can use a compiler to write a compiler, but the quality of the compilers so produced, either in terms of the time required to compile, the memory space requirements or the quality of the object program has been such that these demonstrations have been primarily of academic interest. There has been a tendency to design special purpose source languages whose primary function is the description of the manipulations used in compiling. These have been somewhat more successful. Furthermore, the efforts of the compilers in automatically writing compilers have been primarily limited to the writing of algebraic compilers. It will appear that in the next few years, there will be more need of compilers for the production of data processing compilers than for algebraic compilers. There is every reason to believe that continuing developments in this area, particularly in that of improving the language of a compiler writing compiler will bear fruit before too long.

B. Develop Special Compiler Writing Systems

When one wants to produce a compiler automatically, the circumstances are generally such that no compiler of the type required already exists for the machine in question. Thus the compiler of compilers mentioned above usually operates on a machine foreign to the one for which a compiler is needed and must go through a complicated conversion and bootstrapping routine after compilation of the compiler. A proposal has been to develop a large system in which the inputs are the desired language for the compiler to be produced and the description of the machine on which the compiler is to operate. This is the concept of the SLANG system under development by R. A. Sibley of IBM.

C. Develop Techniques to Facilitate Compiler Writing

At the present time, the writing of a first class compiler is entirely dependent upon the careful fashioning of all sections of the compiler by manual programming symbolic language, instruction-by-instruction. The quality compiler of today will require from 25,000 to 50,000 machine instructions each of which must be hand written and debugged. It has been obvious for some time that, if techniques are available to reduce the amount of coding to be done, it will greatly reduce the compiler writing effort. That is, since the complete elimination of hand coding by using a compiler of compilers is not satisfactory, what is needed is a collection of techniques for minimizing the effort. Let us consider what programming aids have been offered.

1. List and String Processors

Since the nature of most source languages is a rather free running string of meaningful symbols loosely resembling English or mathematical notation, one important task in compiling is the analysis of these strings into suitable origins, delimiters, and terminators and the isolation of the included identifiers resulting (after recognition of their contents) in construction of tables or codes. This suggests that an improved scanning and recognition process would be extremely useful. Among available techniques are the threaded list system of Perlis and associates, the Newell-Simon-Shaw list-processing approach and others.

2. Generalized Analyzers and Generators

Following the scanning and decoding of the raw input, the scanned input is analyzed for its meaning in terms of the whole source program. Holt and Turanski have pointed out that, for a given language, the programming of both the scanning and the analysis is virtually identical no matter what the object machine and the nature of the object program will be. They have therefore suggested the stockpiling of either entire analysis sections or of the logic of analysis sections and making effective use of these in all compilers originating from the same source language. For the actual synthesis or generation of object program, a number of algorithms for producing code from algebraic languages have been developed and published. For the data processing compilers like COBOL, it is customary to use "generators" which produce the desired object program sections. Currently there is fruitful development in the area of generalizing these generators so that they can be tailored to produce desired object coding without re-developing the whole generator for each compiler.

3. Macro Instruction Systems

Another aid to the compiler builder is the use of the most advanced type of macro instructions. It is well known that as assembly programs become more sophisticated and comprehensive, the requirements of the compiler writer become less and less. This may be seen in the ease with which compilers may be written where highly sophisticated assemblers are available. The ALTAC compiler (from FORTRAN language to TAC (assembly) language) for the Philco S-2000 is a good example of this use. Even more advanced macro systems are MICA developed by Owen Mock of North American Aviation and MACROSAP developed by M. D. McIlroy of Bell Telephone Laboratories. An ultimate extension of this concept has been in the development of MOBL by North American Aviation which is a data processing language built entirely of macro instructions which are processed by the MICA system. Using the macro instructions concept, they were able to produce a compiler with a minimum of time and manpower that is capable of translating an excellent data processing language into 7090 symbolic language.

As we have seen, the pressures placed upon those with responsibility in the automatic programming area are enormous. With necessity the mother of invention, it is sincerely hoped that at least some of the many solutions currently being proffered will bring the chaotic situation in automatic programming under control by 1963 or 1964.

A FIRST VERSION OF UNCOL

T. B. Steel, Jr.
System Development Corporation
Santa Monica, California

Summary

UNCOL--UNiversal Computer Oriented Language--is being designed as an empirical, pragmatic aid to the solution of a fundamental problem of the digital data processing business: automated translation of programs from expressions in an ever increasing set of problem oriented languages into the machine languages of an expanding variety of digital processing devices. By application of a program called a generator, specific to a given problem language, program statements in this problem language are transformed into equivalent UNCOL statements, independent of any consideration of potential target computers. Subsequently, without regard to the identity of the original problem language, the UNCOL statement of the problem is processed by a program called a translator, which is specific to a given target computer, and the result is an expression of the original problem solution in the machine language of the desired processor. The advantage of this apparent complication over the current procedure of employing a program called a compiler for direct transformation from problem language to machine language is evident when one examines the number of languages and machines involved and the not inconsiderable expense of translation program construction. If there are M problem languages and N machines, then M·N compilers are required and only M+N generators and translators.

In order to arrive at sensible specifications for UNCOL, certain limitations in its scope are essential. Accordingly, UNCOL is designed to cope with only those problem language and machine language characteristics that can reasonably be expected to enjoy general use in the next decade. Any broader approach shows promise of leading to elegant, impractical results.

A glance at the preliminary specifications for UNCOL shows a language akin to a symbolic assembly language for a register-free, single address, indexed machine. The specific commands are few in number and simple in construction, depending on a special defining capability for the expression of more elaborate instructions. The data referencing scheme is complex, allowing the application of a signed index to the primary address, and permitting both the primary and index parts to be delegated to an indefinite level.

Each item of data, either input or calculated, must be described as to type, range precision and the like by special data descriptive syntactical sentences in the language. These descriptions, additionally, provide information concerning ultimate storage allocation as well as indicators of contextual meaning for the explicit commands.

Supplementary to the instructions and data descriptions are certain declarative sentences, inessential to the explicit statement of the problem solutions being translated, designed to provide information useful to the translator in the introduction of computational efficiency into the object program.

Introduction

One of the harsher facts of life in the data processing world, against which none interested in the effective operation of a digital computing installation dare wear blinders, is the existence of pressures toward diversity. Basic hardware always appears in a variety of shapes, sizes and configurations. New machines arrive on the scene with disconcerting regularity. Since 1952 the number of different types of computers built each year has oscillated about a mean of thirty, and subsequent to 1955 better than sixty per cent of these machines have been commercially built, general purpose devices available to any and all having the inclination and the money.[1] It follows that a problem of steadily increasing magnitude is found in the question of inter-machine translatability of programs.

At the same time, an expanding frontier of applications as well as growing sophistication of machine language has led to the proliferation of the "easy-to-code" problem oriented languages--POLs. Considerable difficulty is attendent to the employment of these languages in view of the effort necessary to maintain an adequate supply of current tools. It is well known that the task of producing translations from problem oriented languages to real machine languages is far from trivial. Until quite recently this translating was done by people--we call this "machine language programming." Now, however, with the exception of a few special cases not relevant to this thesis, the responsibility for performing these translations is being assumed by automation. As a result, requirements appear for processing programs--compilers--that are both expensive and time consuming to produce.

This capital investment in a translation program would be well advised were the concern solely for one problem oriented language and one machine language. However, as indicated above, there is a large variety of machine and problem languages. Thus the required investment increases multiplicity--one compiler for each pairing of problem language and machine language. In addition, the first derivative of development in both machines and problem languages is sufficiently positive that there is a strong tendency toward obsolescence prior to use and the day is not yet with us when programmers will write compilers on lunch hour just for drill. A principal objective of the developments described below is a reduction in the time and money required to live with comfort in this dynamic environment.

UNCOL--UNiversal Computer Oriented Language--is as its name suggests, a language--occasionally referred to by the irreverent as an electronic Esperanto. Conceptually, UNCOL is a linguistic switchbox where problem oriented languages are transformable into UNCOL and, in turn, UNCOL is transformable into specific machine languages. If one is presented with M problem languages and N machine languages, M+N transformations are required in the UNCOL world, while M·N such transformation programs are necessary in the classical case. Elementary mathematical insight shows that when M and N are greater than two it is game, set and match to UNCOL. For the sceptic, Figure I is illustrative of the conventional situation and Figure II the UNCOL schema. A more detailed examination of this position, its rationale and its implications may be found in the literature.[2,3,4]

UNCOL is advertised as a practical solution to the problem described above and, as such, need not be defended against charges that it cannot handle this or that special case--usually one that was designed specifically for the discomfort of the UNCOL builders. A more legitimate indictment is that of unworkability in practice.[5] To date, debate on this matter has generated more heat than light. Indeed, this appears to be a question that is answerable only by appeal to experience. The empirical evidence for a meaningful UNCOL will come from a definition of the language and the subsequent development of transformation programs linking through UNCOL a sufficient variety of problem oriented languages and machines.

It is clear that the initial step in this program is the establishment of a trial language. In order to find a point of departure it is essential to circumscribe the problem and determine a bound beyond which this first version of UNCOL shall not penetrate. To this end a priority class of machines has been selected. This class may be roughly characterized as those general purpose digital computers having a capacity at least as great as an IBM 701 which are currently available or may be expected in a commercial version before 1968. The selection of 1968 as a cutoff point is based on an estimate of the validity of current projections of the characteristics of hardware with which UNCOL must cope. If one attempts to penetrate beyond this point all bets are off on the breadth of applicability of UNCOL as it is now conceived.

Advancement in the problem oriented language technology does not introduce the same difficulty. By designing UNCOL sufficiently similar to the languages of the available computers, it is clear that any problem language for which there is a known compiling algorithm can be put into UNCOL form by a generator program that simply performs this algorithm. If no such algorithm is at hand, then construction of a compiler is impossible in any event and the applicability of UNCOL becomes moot. It follows that the key questions revolve around the feasibility of constructing effective translators for the transformation of UNCOL statements into equivalent expressions in the several pertinent machine languages.

The point of view best adapted to providing an informal exposition of this first attempt at an UNCOL is to consider it in the light of an assembly language for a generalized machine. Thus we shall see that UNCOL has a capability for describing data both syntactically and semantically: i.e., with regard to both structure and meaning. There is a scheme for naming items of data analogous to conventional symbolic addressing and an associated indexing procedure permitting reference to data items standing in some ordinal relationship to a distinguished item. A set of verbs in the imperative mood provides a suitable collection of commands for the language. The objects of these verbs are data names. Verbs are of two kinds, primitive and defined, primitives corresponding to instructions in basic machine language and defined verbs playing the rôle that is assigned in conventional assembly systems to macro-instructions. Finally, there are declarative sentences whose function is the transmittal of information about the algorithm being stated. It is these declarative elements that hold the key to the introduction of efficiency in the object codes by the translation algorithm.

Syntax of Data Description

The internal representations of data vary so widely from machine to machine that any attempt to extract useful common description procedures is doomed to failure. Indeed, the occurrence of both binary and decimal machines precludes an approach of this type. A common factor, however, exists in another direction. All of today's computers, as well as any machine whose construction is sufficiently imminent to be relevant, communicate with the outside world in terms of linear arrays or strings of individual characters. While it is painfully true that no standard character set is in existence,[6] it is equally true that no difference in kind occurs from one set to another. As a consequence, the union over all known and contemplated character sets will yield a satisfactory universe of basic marks.

The guiding principle in establishing this universe of characters is the desire to permit an hypothetical computer to accept input and provide output in the normal orthography of any language, natural or artificial, that is in common use anywhere in the world for scientific, scholarly and commercial purposes, provided there exists the technological capacity to employ such a machine. Practical restraint imposes two limitations which cause the selected set to fall short of this ideal but the only measurable loss is esthetic. Those natural languages that are written ideographically or employ syllabaries must be excluded in order to keep the size of the universe manageable. In the case of mathematical symbolism, a somewhat more ruthless action is called for. Typographical distinction between various levels of subscripting and superscripting must be limited in some manner as it can theoretically go to infinity. The most reasonable course seems to be an insistence on a shift indicator whenever the level is carried beyond the first. Failure to allow any geometry in symbolism is too restrictive and application of some limit above the first seems dreadfully arbitrary.

A synthesis of conversations with printers, (human type), random sampling of mathematical and scientific texts and examination of symbol lists such as are found in large dictionaries, has let to the development of the character set summarized in Table I. Occurrence of more than twenty-six letters in the various roman and italic alphabets is a consequence of the inclusion of compound marks such as letters with accents, the cedilla and the umlaut. One could hardly abbreviate the angstrom unit without the "Å".

<u>UNCOL Character Universe</u>

Alphabet	Number
Roman capitals	55
Roman small capitals	55
Roman lower case	56
Italic capitals	55
Italic lower case	56
Greek capitals	25
Greek lower case	28
German capitals	28
German lower case	30
Cyrillic capitals	35
Cyrillic lower case	35
Hebrew	23
International phonetic	45
Arabic numerals	10
Mathematical symbols	175
Commercial symbols	50
Punctuation	15
Special symbols	35
Grand total	811

<u>Table I</u>.

Allowing three size-position variants of each basic mark--main line, subscript, superscript--a total count of 2433 distinct characters obtains. This is not at all as wild as it might seem at first glance. The author has personally seen some printed outputs from various computers with 196 of the listed basic shapes. It is perfectly feasible to attach a device such as a Verityper to many of today's computers. It would be a good wager that the above set is conservative.

Every finite expression constructed as a linear array of the characters in the UNCOL universe is a conceivable input or output to a program described in UNCOL. In any given case, however, this input and output will be limited to expressions of certain well-defined forms, composed from a subset of the character universe. The nature of these legal forms and the character subset are a function of the <u>problem language alone</u>. In the event that the object computer does not possess the ability to handle the full character subset for either input or output, a transliteration must be established by the UNCOL to machine language translation program, according to predetermined rules, dependent only on the UNCOL character universe and the <u>target computer</u>. Thus, this character universe does for the subproblem of character incompatibility exactly what UNCOL itself purports to do for the general problem of language incompatibility.

UNCOL must have the capability to state definitions of character subsets and legal expression forms in <u>structural</u> terms. In order to accomplish this, a formal language, interpretable as syntax, is required. As a base this language includes the first order predicate calculus with identity.[7] Variables are interpreted as ranging over all finite expressions composable from the UNCOL character universe. Primitive names are specified for each of the characters and the operation of concatenation of expressions is introduced, symbolized by the arch, "⌢". Table II gives a summary of this notation.

$\sim$	denial	not ...
$\vee$	disjunction	... or ...
$\wedge$	conjunction	... and ...
$\rightarrow$	implication	if ... then ...
$\leftrightarrow$	equivalence	... if and only if ...
$\bigwedge$	universal quantification	for all ...
$\bigvee$	existential quantification	for some ...
$=$	identity	... equal to ...
$\neq$	diversity	... not equal to ...
$\frown$	concatenation	... followed by ...

<u>Table II</u>.

The names of the characters are arbitrary. By taking them in some reasonable order, a number can be assigned to each character and these numbers can be used as subscripts to a general symbol to construct graphics for the names of the characters. For illustrative purposes and subsequent use in this paper, let us establish names for some small subset of characters, say a portion of the ALGOL character set. Table III is an example of a set of appropriate names.

<u>Names for some ALGOL Characters</u>

S_0 for "0"	S_{62} for ","	S_{67} for "10"
...		
S_9 for "9"	S_{63} for "."	S_{68} for "#"
S_{10} for "a"	S_{64} for ":"	S_{69} for "+"
...		
S_{61} for "Z"	S_{65} for ";"	S_{70} for "-"
	S_{66} for ":="	

<u>Table III</u>.

Proper combination of the names listed in Table III above with the arch notation gives a mechanism for spelling. For example. the name of the expression "3.14159" in the structural-descriptive form envisioned here is:

$S_3 \frown S_{63} \frown S_1 \frown S_4 \frown S_1 \frown S_5 \frown S_9$

In addition to the ability to express names of explicit inscriptions, use of variables and the logical mechanisms outlined in Table II permits the characterization of kinds of expressions. By way of illustration, let us examine a series of definitions terminating in the explication of a form described in ALGOL as a number.[8] Recalling that variables, which will here be denoted by Greek letters, stand in place of the names of unspecified expressions, we have:

$(\alpha B \beta)$ for $\alpha = \beta \vee \bigvee \gamma (\beta = \alpha \frown \gamma)$, (1)

which is to be read as "the expression α begins the expression β" is defined to be "either α is equal to β or for some γ, α followed by γ is equal to β." Similarly, we have:

$(\alpha E \beta)$ for $\alpha = \beta \vee \bigvee \gamma (\beta = \gamma \frown \alpha)$. (2)

A digit is any one of the signs "0" through "9", so:

$D\alpha$ for $\alpha = S_0 \vee \alpha = S_1 \vee \cdots \vee \alpha = S_9$. (3)

It should be evident that no breach of rigor has occurred through the use of dots in definition (3) as the reader can fill in the missing clauses at will.

An unsigned integer is a string of characters each of which is a digit. Thus we have:

$Ui\alpha$ for $D\alpha \vee \bigwedge \beta (\beta B \alpha \rightarrow \bigvee \gamma (\gamma E \beta \wedge D \gamma))$, (4)

and it follows immediately that an integer is an unsigned integer with or without a prefixed sign.

$I\alpha$ for $Ui\alpha \vee \bigvee \beta (Ui\beta \wedge (\alpha = S_{69} \frown \beta \vee \alpha = S_{70} \frown \beta))$. (5)

A decimal fraction is an unsigned integer preceded by a decimal point, viz:

$Df\alpha$ for $\bigvee \beta (Ui\beta \wedge \alpha = S_{63} \frown \beta)$. (6)

Similarly, an exponent part is

$X\alpha$ for $\bigvee \beta (I\beta \wedge \alpha = S_{67} \frown \beta)$, (7)

and by collecting the parts and performing the obvious, the definitions of decimal number, unsigned number and number follow immediately as:

$Dn\alpha$ for $\bigvee \beta \bigvee \gamma (Ui\beta \wedge Df\gamma \wedge \alpha = \beta \vee \alpha = \gamma \vee \alpha = \beta \frown \gamma)$, (8)

$Un\alpha$ for $\bigvee \beta \bigvee \gamma (Dn\beta \wedge X\gamma \wedge \alpha = \beta \vee \alpha = \gamma \vee \alpha = \beta \frown \gamma)$, (9)

$N\alpha$ for $\bigvee \beta (Ui\beta \wedge \alpha = \beta \vee \alpha = S_{69} \frown \beta \vee \alpha = S_{70} \frown \beta)$. (10)

The above example is indicative of the method one would use in describing the legal forms of data for any problem oriented language. Caution must be exercized in formulating such definition schemes, however. The mechanism outlined above has the capability to describe the syntactical structures of very powerful languages such as the logical language of "Principia Mathematica." In such languages non-constructable items occur and may be syntactically defined. For example, our scheme is capable of defining the notion of non-theorem, clearly non-constructable and even undecidable unless one assumes consistency. In order to circumvent trouble from this source, a set of rules for generator writers must be given which limits the complexity and character of the definitions; rules which the careful framer of definitions will follow instinctively.

Semantics of Data Description

The rather elaborate procedure outlined in the preceding section is sufficient to give the necessary syntactical description of data forms, but it provides no mechanism for introducing the meaning of the defined concepts. It is all very well for a translator to know that a particular juxtaposition of characters is given a specific name, say "number", but this information is of small value unless the translator is also aware of the appropriate mathematical and logical interpretation of this string of marks in whatever contexts it may occur. Provision of this information is accomplished by adjoining a semantic counterpart to the syntactical structure already at hand.

The basic principle involved in this scheme is quite simple. A new primitive operator is introduced which appears only in contexts where it is followed by the name of some expression. This structure is interpreted to mean "the meaning of the following expression." In addition, a notation capable of indicating the various different kinds of data upon which machine instructions can act directly must be provided. As the only types of data involved in computers in the UNCOL priority class are approximations to real numbers and binary patterns, this extra mechanism is relatively insignificant. It is now possible to relate, in a well defined manner, all structures described by the syntax to entities upon which the basic imperatives of UNCOL can operate.

In view of the desirability of having UNCOL versions of the various translator and generator programs, it is important that the semantical mechanism be able to handle the data of these programs which includes statements in the UNCOL language itself. By adopting appropriate rules of formation for the syntactical and semantical statements, it is quite possible to do this and avoid the ambiguities which seem fated to arise. A full explication of these rules is too lengthy for this paper and fragments of the rules are not very meaningful.

Referencing Scheme

The UNCOL referencing scheme--for those who prefer the analogy to a symbolic assembly language, the addressing scheme--is necessarily complicated, including an arbitrary level of delegation or indirectness. This last is essential in view of the existence of both problem oriented languages and machine languages having this kind of capability. For example, if the problem language being generated is a list structure language such as the various Information Processing Languages of Newell, Simon and Shaw,[9] and the elaborate scheme of pointing down a list to find a data item were to be interpreted out in the UNCOL statement of a problem, it would be virtually impossible for a translator to reconstruct this list structure for machine language. In the event that the object machine had the capability of mechanizing some portion of the list structure commands by hardware, the loss in efficiency of the resulting program would be unacceptable. As machines with this character are clearly feasible and under contemplation, UNCOL seems required to shoulder the burden of providing for them.

As many problem oriented languages are designed to handle ordered arrays in a particularly straightforward manner, it is essential that UNCOL have an indexing capability in its referencing scheme. For reasons analogous to those outlined above for multi-level delegation, it is desirable to permit indexing the same freedom.

In order to display the entire scheme some notation is required. We denote basic names by capital roman letters. These serve as names for both primary and index variables, as no distinction is made between these types of entities in data description. We enclose the index name in parentheses and append it as a suffix to the primary. Delegation is indicated by an asterisk.

Delegation of addressing can occur in three basic forms and all combinations of these forms is possible. These are: (1), indexing prior to delegation; (2), indexing subsequent to delegation, and; (3), delegation of the index. If there is no indexing specified, then delegation at the initial level is of the simplest kind. In addition, the delegation may be modified by a level limit, denoted by an appropriate integer following the asterisk indicating delegation. This limit is interpreted to mean the number of levels to which the delegation is carried, independent of the indicated delegation of intermediate names. When delegation has been carried out, willy-nilly, to the indicated level, the delegation process proceeds normally from there unless the original name was marked by a limit modifier, denoted by a prime following the limit integer. In this latter case action is terminated regardless of the status of the name at the given level. Implicit in the above description is the fact that any element to which addressing has been delegated is taken to include all its modifiers as well as the associated index name.

When a string of delegated elements is linked together; i.e., when a delegated element points to a delegated element, which points to a delegated element, etc., a principle of postponement is applied--you step down one more level when required and you do not step back up until everything below is specified. As each name may have an associated index, this leads to a tree searching procedure which terminates only when the tip of each branch is reached.

The following examples will clarify the situation. A(I) means "take A and apply I." A*(I) means "finalize A and then apply I." A(I)* means "take A, apply I, and finalize the result." A*(I)* means "finalize A, apply I, and finalize the result." The index variables themselves may be delegated in the same manner and with the same freedom. Thus, A*(I(J*(K)*)*)* is an allowable combination.

The level limit, which may also be complex in structure, has a clear interpretation. For example, the expression A*5 means "delegate exactly five levels and treat the resulting element as a new address." On the other hand, if a limit modifier is applied, the situation is as follows. A*5' means "delegate exactly five levels and treat the resulting element as the finalized address."

Table IV below gives a set of detailed examples of the UNCOL referencing scheme. In each case, the value called "result" is the ultimate quantity to which the entire schema refers.

UNCOL Referencing Scheme

Name	Value	Name	Value
a	A*	a	A*(I*)*
A	B	I	J
B	result	J	K
		A	B
a	A(I)	B+K	C
I	J	C	result
A+J	result		
		a	Z(V*)*
a	A*(I)	V	U*(K)
I	J	K	P
A	B	U	W
B+J	result	W+P	X
		X	Y
a	A(I*)	Z+Y	T
I	J	T	L*(M)
J	K	M	3
A+K	result	L	N
		N+3	R
a	A(I)*	R	result
I	J		
A+J	B		
B	result		

Table IV.

Imperatives

The fundamental imperative verbs of UNCOL--the instructions of the language--are quite elementary and their meaning will be obvious at once to anyone who has coded for a single address digital computer. It should be observed, however, that no registers of any kind are implied by the command structure, contrary to a conventional machine language, although it would seem at a quick glance that there is at least an analogy to an accumulator. Specification of such a register is necessary only when questions of precision are implicit in the meaning of the command. In UNCOL these questions are answered in the data description sentences, not in the instructions. For ease in studying the situation, however, a generalized and exceedingly pliable accumulator-like gadget may be assumed without creating a difficulty.

An element of context enters into the meaning of the instructions. As is the case with conventional machinery, the instructions are presumed to follow one another in a linear sequence. This is so obvious that it is often overlooked in an analysis of the situation. Because of the capability for modification of data context, it is quite important to keep in mind this sequential flow.

The exact meaning of certain instructions, such as the arithmetic commands, are clearly dependent on the context of the data. Thus a translator for a machine having both fixed and floating arithmetic instructions would have to examine the description of the data called for by the address of the UNCOL command in order to determine which kind of machine language instruction to employ. In addition, each arithmetic command can be tagged with indicators, telling the translator whether the command is to be interpreted as employing the data direct, taking the absolute value (where relevant), or complementing the data first.

With the above preliminaries, the following list of basic UNCOL commands should be easily understood.

TAKE: obtain the value by the address.

ADD: compute the sum (arithmetic or logical as the case may be) of the result of the previous instruction and the value named by the address.

SUBTRACT: compute the difference (arithmetic or logical) between the result of the previous instruction and the value named by the address.

Notice that in both arithmetic instructions the actual operation performed is a function of the data description. In the event that a translator should encounter an instruction and data description combination which implies a mixture of incompatible data, an error has occurred, either in the original problem language statement of the problem or, worse yet, in the coding of the generator. Such a situation would be quite unsafe if UNCOL were being conceived as a language in which humans would code, but as things are it would not seem to be too serious.

The meaning of the commands MULTIPLY and DIVIDE should be evident by analogy with the above descriptions, once the meaning of logical division is explained in some satisfactory fashion. The command REMAINDER permits the gathering of the remainder upon division. It needs no operand.

The command REPLACE is the analogue of the "STORE" instructions of conventional machines. It provides a facility for altering the value which is assigned to a given variable.

All decision making in UNCOL form will be reduced to COMPARE, a command with two objects or operands. These two operands are compared, first to second, and the result of the comparison is remembered--in a real machine this would be by setting a toggle.

In order to implement the branching function, it is necessary to give the UNCOL imperative statements identifiers. These names are, however, distinct from any data names and cannot be used as the operands of arithmetic instructions. No instruction can be modified by an UNCOL statement. Were this permitted chaos would reign. It would become trivial to demonstrate the existence of untranslatable sequences.

Certain instructions--branching instructions of both unconditional and conditional variety--have as their operands the names of UNCOL statements. There are two unconditional branch instructions. The more complex of these, ENTER, is described below. The other, GOTO, is a simple unconditional branch to the statement whose name is the operand of the GOTO statement.

The conditional branch commands are all based on the remembered result of a COMPARE. They are:

IF $<$ GOTO   IF $=$ GOTO   IF $>$ GOTO

IF $\nless$ GOTO   IF $\neq$ GOTO   IF $\ngtr$ GOTO

The meaning of these should be evident.

The ENTER command, together with its complement RETURN, is used for the purpose of transferring control to a subroutine and keeping a record of where the entry occurred. It employs a list, called the "entry list", which must be established by the translator. This list contains the statement names of each ENTER which has been encountered on a last-in-first-out basis. Upon completing the action of a subroutine, a RETURN is given and the appropriate point of return is determined by examination of the last item on the entry list. This procedure permits a subroutine to use itself.

Also used with ENTER are the commands WITH and RESULTS. These follow the ENTER and allow the specification of input and output parameters for subroutines. Use of an additional modifier for the referencing scheme--a relative delegation indicator--permits the delegation of an address relative to the current head of the entry list. Then the list is stepped by one on a temporary basis (for the duration of the address determination) and delegation proceeds. This permits a parameter to be called from the top of a nest of subroutines, which is perhaps recursive, without requiring the asking routine to know anything about the nature of the nest. This is quite important for the proper treatment of certain problem oriented languages.

There are cases, particularly in problem languages designed for writing compilers and the like, where a given data item is operated upon in more than one context by the same program. In order to allow an override of the context that is specified by the data description, an instruction for explicitly determining the context is provided. It is SET CONTEXT and its object is a code specifying context. One of these codes will be called "normal" so that a return to the usual situation is possible.

The UNCOL commands are rounded out by a pair which open Pandora's Box wide! They are DEFINE and END DEFINITION. These serve as a pair of brackets that enclose definitions of complex instructions in basic UNCOL terms. The purpose here is analogous to that found for the use of macro-instructions in conventional assembly programs, but its application is inverted. In the usual case there complex instructions are designed to improve the source language program by making it more compact and easier to construct. In the UNCOL case it is the object language that is the prime beneficiary. These definitions permit a translator to recognize units of program larger than one command and, if the object machine has an instruction for that function, considerable gain in efficiency may be obtained.

Declaratives

Not much can be said about the declarative sentences of UNCOL at this stage. Their general character will be similar to such statements as the following: "The next 20 UNCOL statements form a loop." The information content of these declaratives will be largely a function of the availability of flow information in the problem oriented language. In order to determine the proper kind of statement to include, it is necessary to construct some experimental translators and discover empirically what questions the translator would like to have answered in order to introduce efficiency in the object code.

Work currently under way in an effort to plan and flow chart translators to a variety of different machines with objective of uncovering suitable declaratives. Perhaps six months is not too soon to expect some results.

Recapitulation

In the phrasing of H. G. Wells we observe, "The past is but the beginning of a beginning ..." The version of UNCOL outlined in the preceding paragraphs is the result of extensive cogitation and conversation but its only justification will be the test of workability and that is still to come. Perhaps the whole approach is doomed to failure and oblivion, and then again, perhaps not. One thing at least is sure; Experience and not dialectic will be the judge.

References

1. Bourne, C.P., and Ford, D.F., "The Historical Development, and Predicted State-of-the-Art of the General-Purpose Digital Computer," Proc. of W.J.C.C., 17 (May 1960), 1-21.

2. Mock, O., et al., "The Problem of Programming Communications with Changing Machines: A Proposed Solution," Comm. A.C.M., 1:8 (Aug 1958), 12-18; 1:9 (Sept 1958), 9-15.

3. Steel, T.B. Jr., "UNCOL," Datamation, 6:1 (Jan Feb 1960), 18-20.

4. Steel, T.B. Jr., "UNCOL: The Myth and the Fact," Rev. in Automatic Programming. (in press).

5. Holt, A.W. and Turanski, W.J., "Man-to-Machine Communication and Automatic Code Translation," Proc. of W.J.C.C., 17 (May 1960), 329-339.

6. Bemer, R.W., "Survey of Coded Character Representation," Comm. A.C.M., 3:12 (Dec 1960), 639-642.

7. Hilbert, D. and Ackermann, W., "Mathematical Logic," Chelsea, New York, 1950.

8. Naur, P., et al., "Report on Algorithmic Language ALGOL 60," Comm. A.C.M., 3:5 (May 1960), 299-314, esp. #2.5.1.

9. Shaw, J.C., et al., "A Command Structure for Complex Information Processing," Proc. of W.J.C.C., T-107 (May 1958), 119-128.

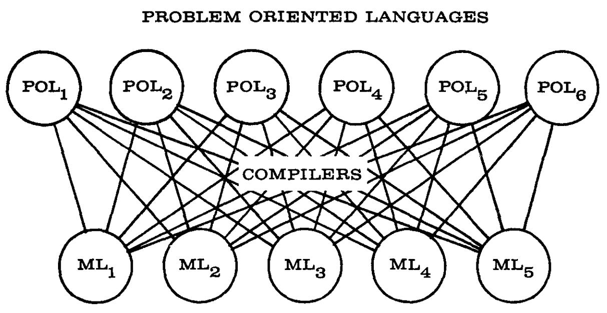


FIGURE I

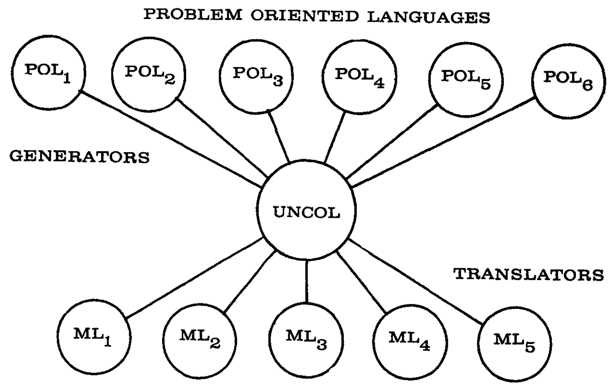


FIGURE II

A METHOD OF COMBINING ALGOL AND COBOL*

Jean E. Sammet
Data Systems Operations
Sylvania Electric Products
Needham 94, Massachusetts

SUMMARY

This paper presents a method for combining ALGOL and COBOL. The purpose and general approach is described; the basic principle is to try to give both groups what they want rather than forcing one group to conform to the other. The major and conceptual differences are listed and described under the headings (a) type of problem to be solved; (b) general usage of the language; (c) symbolism; (d) data description; (e) input-output. Four types and levels of interchangeability are listed and described, namely (a) transliteration of basic symbols; (b) transliteration of groups of symbols; (c) translation of groups of symbols; (d) arbitrary differences.

A listing of elements which are interchangeable under one of the first three categories cited above is given. Arbitrary differences between the two languages are given.

Two sections consider the changes which must be made to each language to have them coincide in the areas where they basically overlap. The paper ends with some indications of the magnitude of the modifications involved.

This paper is intended primarily to show a method which can be used to bring the languages together. The changes can only be made officially through the committee maintaining each language.

I. GENERAL DESCRIPTION OF METHOD

1. Purpose and General Approach

At this point in time, there exist two languages - namely, ALGOL and COBOL - which are designed to be problem oriented languages in the two major fields of computer applications: mathematics, and business data processing. (Some people prefer the terms "procedure-oriented" or "procedural" rather than "problem oriented". Since no definition of any of these terms has ever been agreed upon, the exact wording is not significant for the purposes of this paper.)

Although the need for combining the two languages seems fairly evident, it is worth pointing out a few of the advantages to be gained from such a step. The first - and most important - factor is that of cost. As long as computer manufacturers need to prepare two radically different compilers to satisfy their customers, the hidden "software" costs will continue to be quite high. Even such mundane matters as double training manuals and operating procedures add to the overall price the user pays for his equipment. Since there is a high degree of overlap (as will be shown later) it is certainly reasonable to consider bringing the languages together. A second factor is the desirability of extending the area of standard methods of writing programs. There are certainly large classes of problems for which COBOL is suitable and ALGOL is not, and conversely. A combination of the languages, tentatively named ALABOL (for ALgorithmitic And Business Oriented Language) would widen the class of problems which could be handled in the "standard" way. In no way is this meant to imply that ALABOL would serve as a universal panacea, or even as a single universal problem (or procedure) oriented language. However, experience from using such a language would be useful in narrowing the classical (and still existent) distinction between data processing and mathematical problems.

Both of these languages have many characteristics in common, not the least of which is that they both were developed by committees and have gained a fairly wide acceptance by the computing industry. However, their surface differences are certainly greater than their similarities. The main reason for this is the fact that there is virtually no practical intersection of the two committees. At the time this paper is written, the author is the only person who is officially a member of the two committees charged with the maintenance of the two languages. (This is not meant to imply that there are no other people interested in both languages simultaneously. Many of the people with the

* This is based on the versions available in October 1960 as shown under the references.

dual interest have been unable to participate in both groups.) There appear to be two primary reasons for this lack of connection between the two groups. The first reason involves the basic concept of each language, and the area of problems for which it is intended. In general, those individuals interested in business data processing problems are not interested in mathematical and engineering problems, and conversely. However, the increasing number of problems which cut across both lines makes the amalgamation of these areas more prevalent and necessary. It is becoming much more difficult to retain the old individual compartments for large problems, and this fact must be reflected in language development.

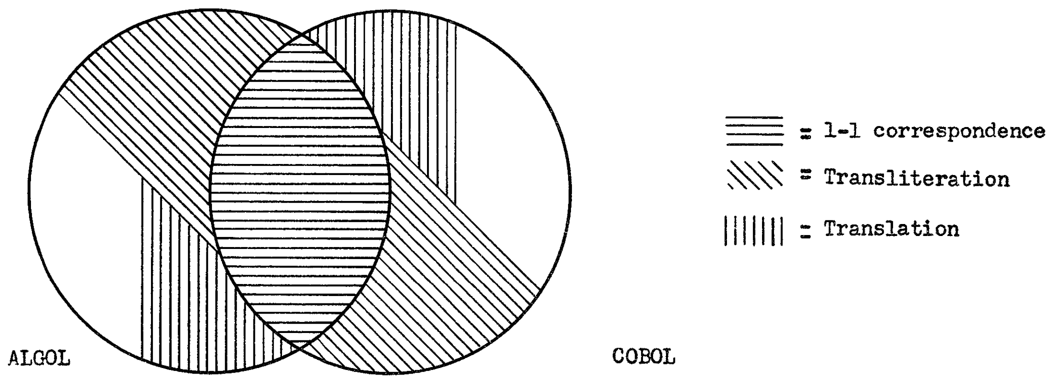
The second reason for the lack of connection between the two groups is a certain amount of displeasure on the part of each with the approach taken by the other. These conceptual differences are indicated in a later section. However, this paper shows explicitly the large amount of common ground which actually exists. Furthermore, the purpose of the work reflected in this paper is to try to give both groups what they want, rather than forcing one group to conform to the other. The result might be considered by some to be a hodge-podge, but the author prefers to think of it as an effective compromise in which the computing industry gains a great deal, and comparatively little unhappiness is generated.

The present relationship between ALGOL and COBOL can be summarized by the following diagram:

agreement than a lack of intersection would normally indicate.

The basic goal of this paper is to show a way in which the essential characteristics of both languages can be maintained. This work is not intended to develop a "universal language", although that might well be a byproduct. The totality of work involved in combining these languages is necessarily split into a few stages. Only the first two are being described here, but the others will be mentioned briefly to show the direction of future work. The first phase involves a clearcut statement of the major conceptual differences. Obviously this is essential if detailed technical work is to be done. The second phase, which is also covered in this paper, involves only those areas of both languages where they attempt to do the same type of operation. Later work will expand the area of concern. The general approach can be summarized by saying that wherever possible use will be made of the types of interchangeability described below in I.2 and significant restrictions, changes, and additions will only be made where absolutely necessary.

In order to allow effective judgment and evaluation of this paper, it must be pointed out that there are several things it does not claim to do. First - and most important - it does not present a complete set of specifications for a new language. In some areas, it has been possible to show all the details, but in others, the magnitude of the work involved is too large to make this worthwhile unless strong support is shown.



The three types of shading represent three types of interchangeability and are discussed in a later section. The actual verbal meaning of the diagram is that there are many points in common between ALGOL and COBOL (where "in common" means that there is a 1-1 correspondence between certain elements). Thus, within the area of intersection, there are at worst some small notational differences which can trivially be resolved. Furthermore, even in the non-intersecting areas, there is a wider range of

Secondly, problems of implementation are not considered here, although nothing has been done to make the lot of the implementor any more difficult.

2. Major and Conceptual Differences

There are five major differences between ALGOL and COBOL. These can briefly be stated as:

(a) Type of Problem to be Solved

(b) General Usage of Language

(c) Symbolism

(d) Data Description

(e) Input-Output

(a) Type of Problem to be Solved

It is probably a truism to state that ALGOL is primarily concerned with mathematical type problems, and in particular, with expressing algorithms. To quote from the ALGOL 60 report, "The purpose of the algorithmic language is to describe computational processes". On the other hand, the COBOL 60 report states that "The task of the committee was that of preparing a common business language. By this is meant the establishment of a standard method of expressing solutions for a certain class of problems normally referred to as business data processing".

It is obvious that this difference in the types of source problems plays the most significant role in any comparison of the languages.

(b) General Usage of the Languages

One of the major points of difference between the two languages is the implied (and sometimes stated) way in which the languages will be used. This is virtually equivalent to the amount of connection with computers each language possesses. More specifically, ALGOL was defined, and is being used primarily, as a communication language. That is to say, the emphasis has been on the establishment of a language which could be used to transmit algorithms and solutions of mathematical type problems. This concept is best illustrated by the use in ALGOL of a reference and a publication language as well as a hardware representation. The emphasis on the first two makes it clear that the actual running of these problems on a computer is considered by many - but not all - ALGOL experts to be a secondary matter. COBOL, on the other hand, has placed a major emphasis on its use in solving specific problems on a variety of computers. That is, COBOL experts are primarily concerned in making sure that the problems can be physically run on more than one computer and get the same answers. The lack of anything but a set of hardware characters (and the limitation of the number of these to 51) is clear proof of the strong computer connection.

It is definitely not the point and purpose of this paper to enter into a discussion on the merits and demerits of either philosophy. It is sufficient to point out that this difference exists, and that it has a major effect in the development of each language.

(c) Symbolism

There are several major differences in symbolism between the two languages. The first of course is the obvious one of having mathematical notation wherever possible in ALGOL, as opposed to the use of normal English wherever possible in COBOL. A second difference involves the establishment of three levels of languages in ALGOL, of which the reference language is the only standard one. The full effect of this is merely a reflection of the point cited in (b) above. Thus, if people are primarily interested in exchanging algorithms, then mundane problems such as non-existent symbols can be ignored. If, on the other hand, the primary purpose is to run problems on a computer, then character sets and hardware limitations must be taken into careful consideration. A third difference is a philosophic one of whether or not the reader should be forced to work very hard. In ALGOL, the emphasis has been placed upon succinctness and the use of symbolism wherever possible. As a result, a person uneducated in ALGOL cannot readily understand any ALGOL program. COBOL, on the other hand, has placed a major emphasis on readability, even to the point of being verbose. A person who has never seen a COBOL manual before can get a fairly good idea of what is going on from a normal COBOL program.

Again, as above, no attempt is being made to sit in judgment on these views, but merely to point them out.

(d) Data Description

The major difference within the area of data description is of course the fact that COBOL has an extremely elaborate system while that in ALGOL is fairly simple. This, of course, only reflects the difference in the primary type of source problem as delineated in (a). To be more specific, COBOL makes heavy use of the concept of a file, whereas ALGOL does not allow for this at all. Furthermore, COBOL requires the user to specify the exact format and placement of both his input and output data, whereas ALGOL only concerns itself with specifying the type of variable. This, in turn, is of course directly caused by the emphasis on the need for a good input-output system in COBOL object programs and an omission of this factor from ALGOL.

(e) Input-Output

To cover the subject of input-output most simply, one needs only say that at least one-third of COBOL is devoted to this problem, whereas it is ignored completely in ALGOL. This is again a reflection of the very basic point given in (b).

3. Types and Levels of Interchangeability

In order to combine these languages with a minimum of change to either, it is necessary to specify the types and levels of interchangeability which exist. There are four of these which are of significance:

(a) Transliteration of Basic Symbols

(b) Transliteration of Groups of Symbols

(c) Translation of Groups of Symbols

(d) Arbitrary Differences

It must be re-emphasized that all references to ALGOL symbols are to the reference language. Actually, the job of combining these languages would be far simpler if one of the hardware representations for ALGOL were chosen, since COBOL is essentially a hardware representation.

The general meaning of (a) - (d) is given here, and Section III contains specific instances of these types of interchangeability.

(a) Transliteration of Basic Symbols

This simply involves a 1-1 correspondence between some of the basic symbols as defined in Section 2 of the ALGOL 60 report, and the character set as defined in Chapter III, 1 of the COBOL report.

(b) Transliteration of Groups of Symbols

In this case, it is possible to establish a particular sequence of ALGOL basic symbols and a particular sequence of COBOL words, because of the fact that a COBOL word is not considered a basic symbol.

(c) Translation of Groups of Symbols

In this case, there is not a 1-1 correspondence, but it is possible to translate certain specific groups of ALGOL symbols into certain groups of COBOL symbols, and conversely.

(d) Arbitary Differences

There are certain differences in the notation which can be easily taken care of by making certain restrictions on one or the other or both languages.

II. DETAILED LISTING OF INTERCHANGEABILITY

Throughout the discussion of interchangeability, only the reference format of ALGOL is used. (See ALGOL report, INTRODUCTION.) Furthermore, no use or consideration is made of optional words in COBOL. (See COBOL report, III, 2.2.3b.)

To save writing, an arrowhead will be used to indicate the direction of the interchangeability. Thus $A \rightarrow C$ means that the specified COBOL characters can be replaced by the designated ALGOL characters. A double-headed arrow means that the replacement can take place both ways.

These lists are not guaranteed to be complete, and in some cases, there are some very subtle points about these correspondences which are not discussed.

1. Transliteration of Basic Symbols

The list (Figure 1) shows the basic symbols in each language which have a 1-1 correspondence with basic symbols in the other language. This correspondence involves the syntax as well as the symbol itself, i.e. they are used the same way. (See ALGOL report, 2, 2.1-2.3 and COBOL III, 1.1 and 1.2).

2. Transliteration of Groups of Symbols

Many of the basic symbols in ALGOL have the same syntactic meaning as full words (which are groups of symbols) in COBOL. The list (Figure 2) actually consists primarily of basic ALGOL symbols and corresponding COBOL words or groups of words. These are considered to be transliterations because a 1-1 correspondence exists between the groups and no translation is required.

3. Translation of Groups of Symbols

It is extremely difficult to list all of the cases in which a group of symbols in one language can be translated into another. In some cases, the correspondence is so complicated as to render a translation either difficult and/or impractical. As an illustration of this problem, consider the existence of the standard functions which are in ALGOL, but not in COBOL. These can be handled either as a subroutine (which does not allow the programmer to ever write the actual form) or by means of the DEFINE. In the latter case, it is rather a moot point as to whether or not is is possible to mechanically translate the following COBOL statements into sin (X):

```
DEFINE VERB sin AS COMPUTE Z =

(appropriate formula for computing

sin (X) ) WITH FORMAT sin (X) = Z.
```

In order to computer A = $\frac{1}{2}$ sin (Y) one would have to write

```
SIN (Y) = W

MULTIPLY W BY .5 GIVING A
```

This point is mentioned again in Section III below.

With the realization that not all "translatable" groups can be shown, it is still possible to give some widely differing illustrations of cases where this is possible and practical. This is done in Figure 3.

4. Arbitrary Differences

The question of whether or not the difference between two expressions is "arbitrary" is certainly a matter of opinion and calls for a specific value judgment. Rather than stir up a storm on this minor issue, it seems better to say simply that any case in which a straight two-way transliteration is possible is considered arbitrary. Under this very restrictive definition, it should be noted that the only "arbitrary difference" is in the use of the symbols, * and X for multiplication in COBOL and ALGOL respectively, and the use of ÷ as a division operator in ALGOL.

III. DEVIATIONS OF COBOL FROM ALGOL

This section concerns itself with the ways in which COBOL deviates from ALGOL, considered only for the areas in which they basically coincide or overlap. (This same procedure is reversed in the next section, where COBOL is considered as the base.) Thus, the main discussion centers around the PROCEDURE DIVISION, with some comments on, and additions to, the DATA DIVISION.

The material shown here involves comparisons as well as associated suggestions and comments for changes which are needed to make COBOL contain equivalent capability to ALGOL. The general approach taken in both this section and the next is to add the necessary features to each language - without destroying its inherent characteristics - rather than placing restrictions. Thus, additions are made to each language within its own framework, so that equivalent capability is available although in a somewhat different form.

In order to avoid repeating material, both sections III and IV must be considered simultaneously if a complete comparison is desired.

1. CHARACTERS AND WORDS

a) Word Formation

COBOL differs from ALGOL by allowing purely numeric procedure-names, by allowing the letter in data-names to appear anywhere, and by restricting all words except literals to being less than or equal to 30 characters. The first two of these items (i.e., data and procedure-name differences) are somewhat arbitrary although there are good reasons for the choice made in each language. The restriction on the maximum number of characters for words is desirable for implementation reasons, although the number 30 is somewhat arbitrary.

One minor difference is that in COBOL, two procedure-names are equal only if they are identical, whereas in ALGOL, leading zeroes are ignored.

COBOL does not have any means of representing numbers in their floating point form. The ALGOL form cannot be adopted because of the lack of a suitable character for the "10". However, it is easy to add the concept in the DATA DIVISION.

b) Subscripts

Subscripting in COBOL is much more restricted than in ALGOL. From the point of view of the language, it is very easy to allow the more general form. Thus, to have equivalents in the two languages, COBOL subscripting must first be extended to allow any number; i.e., remove the restrictions to 3 subscripts. Then COBOL must allow subscripts to be subscripted themselves, and finally, must allow any arithmetic expression to be used as a subscript.

c) Functions

The ability to define arbitrary functions and the standard functions should both be added to COBOL. Note however, that it is not actually necessary to include this capability directly in the specifications because the same logical results can be achieved by using the DEFINE. The only disadvantage to this is the awkwardness of the method. (See II.3 above.)

2. PROCEDURE DIVISION

a) General Procedural Syntax

COBOL currently allows only 2 hierarchies of named procedures; i.e., paragraphs and sections. To cope with the more general ALGOL form, it is necessary to allow any number of nested named procedures in COBOL. This of course should be done by adding BEGIN and END with the same definitions as in ALGOL. Note that COBOL already contains "sentences" which are the logical equivalent of ALGOL compound statements. It is not necessary to introduce declarations into the PROCEDURE DIVISION since the equivalent information can be imbedded into the DATA DIVISION.

Since COBOL conditional statements and sentences* contain the ALGOL form as a special case, no change is needed. Furthermore, the basic verb structure can be kept; since it is shown below that ALGOL operations can be expressed in terms of COBOL verbs.

ALGOL procedures can be handled equivalently through the use of the DEFINE.

b) Arithmetic & MOVE Verbs

The five COBOL arithmetic verbs, ADD, SUBTRACT, MULTIPLY, DIVIDE, and COMPUTE, as well as the MOVE, can be used to express ALGOL assignment statements. It is necessary to extend the capability of each verb to handle floating point numbers.

c) Procedure Branching

The simple GO (i.e., Option 1) of COBOL is of course equivalent to the go to of ALGOL. The switch declaration in ALGOL can be handled by the proper combination of GO TO ... DEPENDING ON, ALTER, and computational procedures.

The COBOL PERFORM is essentially a special case of the ALGOL for. However, the PERFORM must have several options added to it to give it the full power and generality of for. In particular, in the TIMES option, the ability to show any number of these in a single PERFORM must be added. The step until can be handled provided the VARYING option of the PERFORM is allowed to apply to any field and several VARYING's can be written in one PERFORM. The UNTIL and while are simply logical negations of each other. Thus, the PERFORM can be extended easily to make it equivalent to the ALGOL for.

The NOTE of COBOL and the comment of ALGOL are equivalent. Furthermore, the use of an extra semicolon to show a dummy statement in ALGOL is equivalent to EXIT in COBOL.

3. DATA DIVISION

The major additions to the COBOL DATA DIVISION are to the CLASS clause in the Record Description. The additions will then result in the equivalent of the type declarations. Allowing BOOLEAN to be specified in COBOL will cause the meaning to be identical in the two languages. Any data-name which is NUMERIC and does not have a decimal point is an ALGOL integer; all other NUMERIC fields are real in the ALGOL sense. Finally, a FLOATING-POINT category must be added.

* Defined in TC 68.1 - See Reference 2.

It should be noted that the information given in an Array Declaration is basically available in the OCCURS clause and the level structure of the Record Description.

IV. DEVIATIONS OF ALGOL FROM COBOL

This section is similar to III in that it deals with deviations of the two languages. Therefore, the general remarks at the beginning of III apply fairly well here, except of course that the changes will be suggested for ALGOL to give it equivalent capability to COBOL in the areas where they are inherently similar. This latter qualification is extremely important, because no attempt is being made to add full description of data, input-output, or environment to ALGOL.

In order to avoid repeating material, both sections III and IV must be considered simultaneously if a complete comparison is desired.

1. BASIC SYMBOLS AND CONCEPTS

a) Letters

The use of both upper and lower case letters cannot be handled in COBOL for hardware reasons. Therefore, the use of the lower case letters in ALGOL should notbe allowed.

b) Identifiers

The requirement for a letter at the beginning of each identifier should be removed. Furthermore, labels can be allowed to be purely numeric since they can be identified from the context. A restriction on the length of any identifier to 30 characters seems reasonable when implementation is being considered.

c) Variable Types

Although ALGOL does not have all the variable types that COBOL does (e.g. ALPHABETIC) it does not seem meaningful to add this ability since there is no real way to handle it, and adding this to ALGOL is inherently dependent on large extensions in the area of data description.

2. SYNTAX

a) Conditional Statements

The primary addition to ALGOL needed to cover the COBOL syntax is the extension of conditional statements to allow nested "if clauses". This form is one of several proposals that has been made to handle the ambiguity which now exists in ALGOL. (See 4.5 in ALGOL report.)

b) for Statements

The main change needed here is to add the ability to apply the for to procedures other than those immediately following it.

3. COBOL VERBS

Because of the importance of the verbs in COBOL, it is desirable to discuss explicitly their ALGOL counterparts.

a) Arithmetic and Data Movement

Some of the COBOL verbs appear on the surface to have no ALGOL equivalent, but this is not really true. As pointed out in III, the five arithmetic verbs and the MOVE are equivalent to assignment statements. The EXAMINE which appears to have no direct counterpart in ALGOL actually does not need one, since the EXAMINE is not primitive. That is, the EXAMINE can always be expressed as a proper combination of PERFORM, equality tests, and MOVE. Since these can always be put into correspondence with ALGOL concepts, the EXAMINE is then just a summation of these correspondences.

b) Procedure Branching Verbs

See III, 2(c) above.

c) STOP Verb

The ability to signify the stopping point of a program does not appear in ALGOL. This can be handled either by modifying the end or by just adding a separator stop.

d) Input-Output Verbs

There is obviously no ALGOL equivalent for this.

e) Compiler Directing Verbs

The ENTER is of course equivalent to a procedure so there is no problem here. The EXIT is equivalent to the use of an extra semicolon in ALGOL. The USE and INCLUDE have no direct counterpart in ALGOL since there is no input-output of any kind, nor any reference to a library. However, the same effect can be obtained through the proper use of procedures. The DEFINE is essentially equivalent to a procedure. Finally, the NOTE is, of course, equivalent to comment.

4. DATA AND ENVIRONMENT DESCRIPTION

For the reasons given in I.1 above, these COBOL elements are not existent in ALGOL, except for a few special cases which are handled by type declarations. Files as such cannot be described in ALGOL. Similarly, ALGOL contains no provision for describing any of the environmental conditions under which the problem is to be run on a computer. It is beyond the scope of this paper to consider this type of addition to ALGOL.

V. COMMENTS AND CONCLUSIONS

This paper has chosen a method whereby ALGOL and COBOL (as defined in the references) can be combined to produce a new language called ALABOL, which will contain major portions of ALGOL and COBOL as subsets. This work has been done under the strict - although self-imposed - requirement that neither language should have its basic structure altered in order to conform to the other. Because this was the basic approach taken, the resultant combination is perhaps less powerful than might otherwise be achieved. On the other hand, it is a very practical fact of life that the circumstances under which both of these languages were created do not lend themselves kindly to major conceptual changes. It is the strong opinion of the author that the additions (and changes where necessary) are definitely not major. It seems that if both the maintenance committees really wish to take advantage of the power to be gained by this method, then this can be done fairly easily. None of the changes or additions appear to cause any large amount of difficulty; in fact, some of them have already been discussed as "something to be done in the future".

As was stated earlier, this paper definitely does not intend to present the final specifications for a new language. What it has done is point out quite specifically just what changes and additions are needed to bring the languages together under the ground rules previously discussed. In most cases, full details have been given, whereas in some other instances, some of the details are available but not shown in the paper. It should also be noted that this paper did not include the addition to ALGOL of input-output, full data description, or environmental descriptions to ALGOL. This would have involved a very major conceptual change in the structure of the language and, therefore, would violate one of the current ground rules. It is expected that this will be done in the future.

The method shown here is practical and not difficult to apply. It should be of interest to users and manufacturers, to both members and non-members of the maintenance committees, and to all members of the computing industry who have any interest in the development of powerful procedural and problem oriented languages.

REFERENCES

1. Report on the Algorithmic Language ALGOL 60, edited by Peter Naur, Communications of the Association for Computing Machinery, Vol. 3, No. 5, May 1960.

2. COBOL: Report to Conference on Data Systems Languages, April 1960, U.S. Government Printing Office #1960 O-552133 plus 3 supplements distributed to CODASYL mailing list, plus change labeled TC 68.1 which has been officially approved by the maintenance committees.

ALGOL		COBOL	Comments
0,1,...,9	←→	0,1,...,9	Identical symbols.
A,B,C,...,Z	←→	A,B,C,...,Z	Only upper case letters are allowed in COBOL.
+	←→	+	Identical symbols.
-	←	-	Used to indicate subtraction. Since this character has two uses in COBOL, the arrow cannot go both ways.
X	←→	*	
/	←→	/	Identical symbols.
÷	←→	/	
<	←→	<	Identical symbols.
>	←→	>	
=	←→	=	
,	←→	,	
.	←	.	This is the use as a decimal point in numbers, and does not involve the use as a sentence terminator in COBOL. (See <u>end</u> below.)
;	←→	;	Identical symbols.
#	←	Space	# is used only in ALGOL strings and so the arrow can go only one way.
[	←	(	Used as subscript delimiters. The use of parentheses for arithmetic expressions in COBOL prevents the arrow from going both ways.
]	←	)	
(	←	(	Used in arithmetic expressions.
)	←	)	
‘	→	"	The pair ‘ , ’ serve as string delimiters in ALGOL which are similar conceptually to literals in COBOL. Because ALGOL uses two symbols, there is not a full double correspondence.
’	→	"	
end	←	.	The use of the decimal point for numbers in COBOL prevents the double correspondence.

<u>Figure 1</u>

ALGOL		COBOL	Comments
$=$	$\longleftrightarrow$	EQUALS	ALGOL basic symbol, COBOL word
$=$	$\longleftrightarrow$	EQUAL TO	ALGOL basic symbol, group of COBOL words
$\geqq 0$	$\longleftrightarrow$	POSITIVE OR ZERO	Groups of symbols in both languages
$\leqq 0$	$\longleftrightarrow$	NEGATIVE OR ZERO	
>0	$\longleftrightarrow$	POSITIVE	ALGOL group of symbols, COBOL word
<0	$\longleftrightarrow$	NEGATIVE	
$>$	$\longleftrightarrow$	IS GREATER THAN	
$>$	$\longleftrightarrow$	EXCEEDS	
$\uparrow$	$\longleftrightarrow$	**	ALGOL basic symbol, group of COBOL symbols
A [X,Y]	$\longleftrightarrow$	A(X,Y)	Groups of symbols in both languages
<u>comment</u>	$\longleftrightarrow$	NOTE	ALGOL basic symbol, COBOL word
<u>go to</u>	$\longleftrightarrow$	GO	ALGOL basic symbol, COBOL word
$\vee$	$\longleftrightarrow$	OR	ALGOL basic symbol, COBOL word
$\wedge$	$\longleftrightarrow$	AND	ALGOL basic symbol, COBOL word
$\neg$	$\longleftrightarrow$	NOT	ALGOL basic symbol, COBOL word

Figure 2

ALGOL		COBOL	Comments
$7.2_{10}2$	$\longleftrightarrow$	720	This is just one example of the whole class of ALGOL representations of numbers written in exponential form.
Y: = X-5	$\longleftrightarrow$	COMPUTE Y = X-5	This is, of course, just an illustration since the COMPUTE verb is equivalent to the := in ALGOL.
<u>for</u> x: = 1 step until n	$\longleftrightarrow$	PERFORM ... n TIMES	This is again an illustration.

Figure 3

ALGY - AN ALGEBRAIC MANIPULATION PROGRAM

M. D. Bernick, E. D. Callender and J. R. Sanford,

Western Development Laboratories - Philco Corporation - Palo Alto, California

Summary

In a great variety of scientific problems, the reduction of complex analytical expressions is highly desirable but often such reductions, although straight forward, are extremely lengthy and laborious. The following paper describes a program written for a high-speed digital computer which accepts algebraic expressions as input and outputs a similar set of modified expressions. The description includes the definition of the ALGY operations used to obtain the desired results, followed by an explanation of the logical flow of the program, and concluding with a description of future operations which will be incorporated into the system.

Introduction

The kinds of problems which initiated interest in general purpose high-speed digital computers were, for the most part, problems which involved an extreme amount of arithmetic. Without computers, in many cases, no attempt could be made to solve them, as valuable as their solutions were deemed to be, not only because of the time required to perform the computations, but also because of the very small probability that the results after months of hand calculation would be correct. With the advent of the electronic computer, the arithmetic involved in these problems became a trivial matter and solutions were easily effected with a high degree of reliability.

Recently, an analogous difficulty has arisen in solving another kind of problem. We wish to solve systems of differential equations by perturbation methods. These problems, rather than involving arithmetic, required an overwhelming amount of algebraic manipulation. The only feasible way to handle this kind of problem is, again, to "let the computer do it", and it was for this purpose that the computer program called "ALGY" was developed.

ALGY is an interpretive routine through the use of which the programmer or mathematician may instruct the computer to perform certain algebraic manipulations. ALGY is basically an elaborate scheme for the manipulation of alphanumeric bit patterns. The input and output used with ALGY are alphanumeric algebraic statements. In this paper, we will give the definitions pertinent to the ALGY program, describe the currently available algebraic manipulations in a brief description of the flow of the ALGY program, demonstrate by use of an example an ALGY program, and, finally, give a brief outline of future algebraic operations and applications.

Definitions

In order to understand the manipulation which ALGY performs, it is necessary to define a few special terms. The basic building block in ALGY is the BCD character. These characters are combined in various ways into quantities. There are three types of quantities in an ALGY program: numeric, such as 51/725; special, such as plus, minus, period, parentheses, dollar sign, asterisk and quasi-alphabetic, such as cos 5X. Because it is desirable in algebraic manipulation to do exact arithmetic, all numbers are represented as fractions. The restriction on the numeric quantity is that the total number of decimal digits of the numerator and denominator must be less than or equal to 15. Under this restriction, arithmetic operations are exact. The quasi-alphabetic quantities, also, must have less than 16 characters in them and they must begin with a letter. Those restrictions are mechanical ones and could be removed by going to multiple arithmetic precision and using larger storages. Of course, in both numeric and quasi-alphabetic quantities, no special symbols can appear.

A group is an algebraic expression contained within a plus, minus or a parenthesis. It may contain several quantities, for example

```
-13/25*X$2*sin2X*Coef
```

is a group.

An expression is any algebraic combination of groups and/or quantities which are terminated by a period. Expressions are tagged with a name. This name may also be a quantity in another expression.

Throughout the remainder of this paper, reference will be made to different programs, the ALGY system and particular ALGY programs. The mathematician writes the ALGY program which is processed by the ALGY system. ALGY does not manipulate equations, but only expressions. However, it is easily seen that for algebraic manipulations, expression manipulation is sufficient.

ALGY Operations

There are a few general restrictions on the size and type of algebraic expression that ALGY can handle. Any expression can contain at most 4500 BCD characters. Also, all exponents must be positive integers and the only available symbols are the usual 64 BCD characters.

The currently available basic ALGY commands are EQAT, INQT, BUGG, OPEN, SBST, FCTR, TRGA and DONE. When the ALGY system was being designed, we felt that these commands constituted a minimal system to perform the algebraic manipulations that we were interested in performing. Following is a brief description of each command. It is very easy to learn to use ALGY. Two hours of instruction are all that is usually required.

EQAT:
Equate merely records on tape the left and right hand sides of the equation. The algebraic expression on the right hand side of the equate symbol is always preceeded by its name, that is, the symbol used on the left of the equate symbol. This method uniquely determines all expression recorded on the tape.

INQT:
Internal equate renames an expression already on tape, preserving the recorded name. This allows an algebraic expression to be re-equated without having to write the whole expression over again.

BUGG:
Bugg is essentially an "unequate" operation. It searches the tape for the tag that is to be "bugged" and deletes it. This is useful if the user wishes to define a particular variable several different ways during a single ALGY program.

OPEN:
Open removes parentheses from an algebraic expression. To do this, it performs all algebraic multiplication necessary, grouping identical terms, and sorting in a quasi-alphabetical manner.

SBST:
SBST substitutes one or more expression in a given expression. The routine inserts parentheses about each expression substituted.

FCTR:
FCTR factors a given expression with respect to a single variable, which may be exponentiated with the option of equating its coefficient to a given symbol for future reference. The expression may be factored with respect to several variables with the restriction that if a particular group contains two or more variables to be factored, it can be factored with respect to only one of the variables.

TRGA:
Trig A expands a product of sin and cos functions of a given argument to a sum of sin and cos functions of multiple angles. An exponentiated sin or cos function would fall under this category.

DONE:
Done is a control word which allows several independent problems to be processed during the same run.

Logical Flow of ALGY

ALGY accepts algebraic expressions as input, processes these expressions in the manner in which the user has programmed, and outputs a similar set of expressions. To accomplish this, a logical sequence of events is followed.

The algebraic equations are coded in the ALGY format which in essence is simply rewriting them on coding sheets, using English letters instead of Greek symbols when applicable, followed by all the ALGY operations necessary to obtain the desired results. After the program is key punched it is submitted for recording on tape.

ALGY accepts the coded statements, printing and performing all operations directed by the input program. A very simple example follows to facilitate explaining the logical flow of ALGY. Consider

$$e(f,g) = (fg + 1)^3$$

$$f(x) = 1 + Ax + 1/2\ A^2x^2 + 1/6\ A^3x^3$$

$$g(x) = Bx - 1/6\ B^3x^3 + 1/120\ B^5x^5 - 1/5040\ B^7x^7$$

and suppose the factors of x^{11}, x^9, x^7, x^4 and x^3 of the function e are desired, neglecting all terms of order greater than 11. The ALGY input program would appear as follows:

```
EQAT E = (F*G + 1/1)$3.  ie(Equate E = (FG + 1)³).

EQAT F = 1/1 + A*X + 1/2*A$2 + 1/6*A$3*X$3.

EQAT G = B*X - 1/6*B$3 + 1/120*B$5*X$5
         - 1/5040*B$7X$7.

SBST E/F,G.  ie(Substitute in E, the expressions
         equal to F and G.

OPEN E.  ie(Remove all parentheses).

FCTR E/X$11,E11/X$9,E9/X$7,E7/X$4,E4/X$3,E3/,ER.
         ie(Factor E with respect to X$11 and
         call the coefficient E11, factor the
         remainder of E with respect to X$9
         calling its coefficient E9, etc., tag-
         ging the remainder of E after all fac-
         toring is completed, ER)

EQAT X = 0/1.
SBST E11/X.
OPEN E11.
DONE.
```

ALGY will accept the first three EQAT commands, print and store the algebraic expressions on tape, where each expression is identified by its tag on the left-hand side of the equate symbol. It then accepts the SBST command, searches the tape for the expression equal to E and reads it into memory. The routine locates and reads the expression equal to F, and examines E, substituting the F expression with parentheses around it each time it appears in E. After F has been substituted, the same procedure is done for the expression G. The resulting substitution in E would appear in print as follows:

E = ((1/1 + A*X + 1/2*A$2*X$2 + 1/6*A$3X$3)*(B*X

- 1/6*B$3*X$3 + 1/120*B$5*X$5

- 1/5000*B$7*X$7) + 1/1)$3.

ALGY accepts the OPEN command and commences removing the parentheses in the E expression above. The results are printed and stored on tape. The system then enters FACTOR which will factor E with respect to each variable requested. It then prints and stores the coefficients of each factor as well as the new factored E expression as follows:

E11 = E11(X^n,A,B) where n assumes all integers 1 through 19

E9 = E9(X,A,B)

E7 = E7(X,A,B)

E4 = E4(X,X^2,A,B)

E3 = E3(A,B)

ER = ER(X,X^2,A,B)

E = E11*X$11 + E9*X$9 + E7*X$7 + E4*X$4

+ E3*X$3 + ER.

To eliminate all higher order terms in E11, simply EQAT x to zero, SBST x into E11 and then, E11 = E11(A,B).

In this manner the coefficients of X^{11} and X^3 can be accurately determined with a minimum of effort on the part of the user. The computer cost of the solution for a few typical problems is less than 1/6 the cost of the solution obtained by manual labor, assuming that the man performing the algebraic manipulations is as reliable as the computer.

Below is a flow diagram for the ALGY system. Because of the logical complexity of each subroutine, detailed flow charts have not been included in this paper.

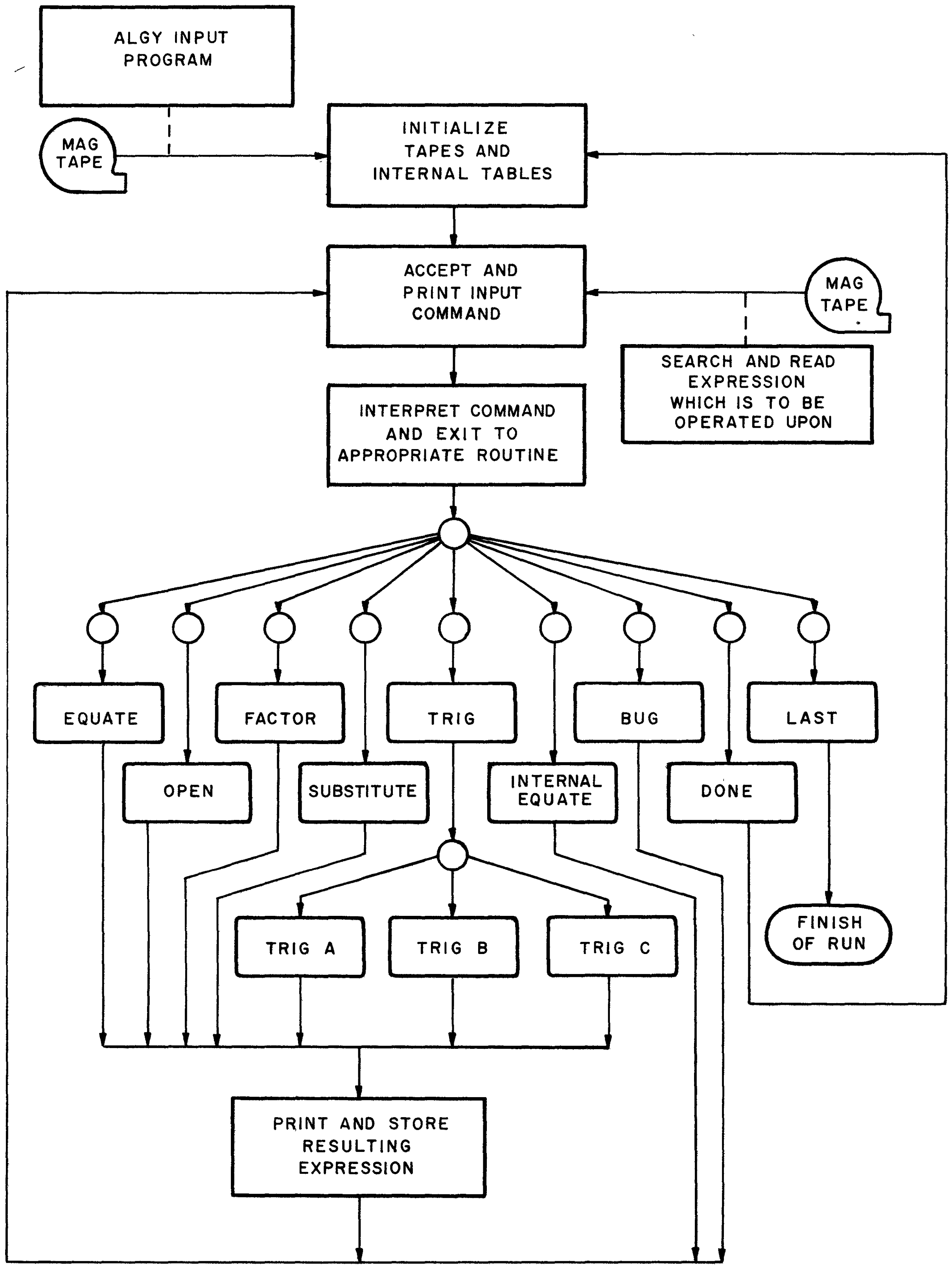
Future Operations

We will tremendously increase, during the next few years, the scope and number of allowable ALGY commands. Because of the specific problem in perturbation theory for which ALGY was originally designed, we will immediately develop two more trigonometric manipulations called TRIG B and TRIG C. TRIG B allows angle variables to be separated using the laws of addition for the sin and cos functions. TRIG C is, in a certain sense, the inverse operation to TRIG A. In TRIG C, sins and cos of multiple angles are reduced to powers of the sin and cos of the angle.

There are many algebraic operations that suggest themselves now that we have the basic tools for algebraic manipulation developed. Our future plans include differentiation, restrictive forms of integration and solutions of linear systems using determinants. The restriction on the exponents will be lessened. We have already found several cases where "super" ALGY commands would be of value. For instance, it would be extremely desirable to have a command to generate a polynomial of given degree when only the ith term is given. There are also numerous form of factorization which suggest themselves. It should also be noted that the ALGY coded program is a straight-flow program just as the first numerical programs were. Undoubtedly, we will develop a loop technique for the ALGY system.

Conclusion

ALGY represents a first step in a new form of computer usage. ALGY is not a general problem solver. If the mathematician does not know how to algebraically manipulate his equations, ALGY can be of little help. But ALGY is an extremely powerful tool in the hands of an intelligent user. It enables the mathematician to consider and to solve problems that he would otherwise never consider because of the large amounts of algebraic manipulation necessary for a solution. It enables him to try different forms of a solution and use different approaches to the same problem, where before he was often committed to just one approach because of the large amount of time necessary to verify that one method.


ALGY INPUT PROGRAM
MAG TAPE
INITIALIZE TAPES AND INTERNAL TABLES
ACCEPT AND PRINT INPUT COMMAND
MAG TAPE
SEARCH AND READ EXPRESSION WHICH IS TO BE OPERATED UPON
INTERPRET COMMAND AND EXIT TO APPROPRIATE ROUTINE
EQUATE
OPEN
FACTOR
SUBSTITUTE
TRIG
INTERNAL EQUATE
BUG
DONE
LAST
TRIG A
TRIG B
TRIG C
FINISH OF RUN
PRINT AND STORE RESULTING EXPRESSION

A NEW APPROACH TO THE FUNCTIONAL DESIGN OF A DIGITAL COMPUTER

R. S. Barton
Computer Consultant
Altadena, California

Summary

The present methods of determining the functional design of computers are critically reviewed and a new approach proposed. This is illustrated by explaining, in abstracted form, part of the control organization of a new and different machine based, in part, on the ALGOL 60 language.[1] The concepts of expression and procedure lead directly to use of a Polish string program. A new arrangement of control registers results, which provides for automatic allocation of temporary storage within expressions and procedures, and a generalized subroutine linkage.

The simplicity and power of these notions suggests that there is much room for improvement in present machines and that more attention should be given to control functions in new designs.

Introduction

The ideas presented arise from the conviction that for a true general purpose digital computer both coding and operation should be fully automated. Higher level programming languages, such as ALGOL, should be employed to the practical exclusion of machine language; questions of efficiency of object program and translation process ought not to arise if the machine has been properly designed. Operation should be under the control of the machine itself, in a fuller sense than is typical in current practice. The functions of scheduling, segmentation of programs for multi-level storage, and control of input - output operations should be handled by a general operational program.

This new approach will be illustrated after reviewing the customary methods of machine design.

The Special Purpose Machine

In simple and well defined applications, the design engineer may dispense entirely with programming assistance and the program may be entirely, or in large part, in the hardware. If the processing required is complex, programmers are invited to assist the engineers. There will be a period of trading-off programmed and component logic, but the resulting machine will tend to resemble the conventional general purpose computer.

The Engineers' General Purpose Machine

In the design of machines to meet competition, the utilization of new components is likely to be of vital concern to the designers. While requiring a complete new set of programs, the new product seldom shows more than minor variations of the traditional design. Its new features originate with both programmers and logical designers, but those ideas which are contributed by the programmers stem usually from applications experience with previous machines rather than from systematic theory. The logical designer has the last word and is most likely to accept ideas which require a minimum of new design.

The Programmers' General Purpose Machine

It seems to be the case that as yet no machine's design has been significantly effected by persons experienced in the development of automatic programming systems. Programming still must be imposed upon designs that have been determined by marketing pressures and tradition. It must be admitted that the programmers of the last decade have been poorly prepared to make the kind of contribution that should be expected. The art of programming has developed in a helter-skelter manner, leaving behind little of value. There is almost no theory and little standard methodology. The logical designers have a large body of switching theory as a basis for their work and have, consequently, done better.

The Simultaneous Design of Computers and Programming Systems

Rather than hope for a new spirit of cooperation among disparate product planning, engineering, and programming departments, a single small organization is needed for each product conception. This would be comprised of three kinds of people responsible for aspects of system model, program design, and logical design.

As an example, for a computer to process applications expressible in the ALGOL 60 language, the system model group would interpret the language, specify a hardware representation and necessary language supplements, define speed or cost objectives, and the "use image" the machine is to present. They would have responsibility for ensuring proper interpretation of the model by both programmers and logical designers.

The program designers would have background experience in the construction of translators and some knowledge of logical design. They will consider necessary reductions in the source language and translation techniques to enable efficient object time interpretation by hardware.

The logical designers would be oriented in current programming practice and become familiar with the source programming language to be used. Their task is to produce designs to handle the reduced languages.

Concepts for the Design

Some simple ideas are now presented that arise quite naturally from using the ALGOL 60 language as a model. These ideas have actually entered into the design of a new Burroughs Information Processing System. For purposes of exposition, they will be considered out of the context of the actual machine, and liberties will be taken to avoid complications which are not germane to the subject.

Polish String Program

The Polish notation was invented by the logician Lukasiewicz for use in the propositional calculus. It has the advantage that rules of operator precedence and signs of grouping are not required. At least two logical machines have been built which use it as a source language. The first of these was the Burroughs Truth Function Evaluator[2]. More recently, Bauer described a similar logical device[3] and hinted of remarkable results which were discussed in another paper[4]. During the past few years, numerous persons have "discovered" the extension of this notation to arithmetic expressions and found it useful as an intermediate language in designing algebraic translators.

Any expression, as defined by ALGOL 60, can be simply translated into Polish form, and this can be the basis of an efficient machine language to be used in place of the customary single or multiple-address command formats. Methods for doing this are well known and familiarity with an algorithm is assumed. While it would be quite feasible to construct a machine to directly interpret ALGOL expressions having suitably restricted identifiers, it was decided, in view of the simplicity of the transformation, to use the Polish form. At this point, it is important to stress that programs need never be written in Polish form; and no lower level assembly-type language is required.

This form provides a machine language with many desirable properties. Programs consist of a string of elements which correspond to identifiers, literals, or operators. All the operators defined in ALGOL 60 can be included. Using the stack construct, next described, there is no need for store and fetch commands such as are normally associated with the temporary storage of intermediate quantities in the evaluation of expressions. More important, the resulting program is in a better form for the application of procedures to improve program efficiency than is the case with the usual command languages.

The Stack Construct

Assume that the elements of the string are examined from left to right so that operators need not be deferred. Normally, for unary and binary operators, the transformation to Polish form would ensure (excepting in the case of procedures) that at most two operands would have to be fetched before each execution.

To mechanize the Polish string program, a special address-length register called the stack counter is provided to hold the most current cell address in a vector of temporary storage cells referred to as the stack. Two word-length arithmetic registers hold the most recently fetched or computed operands. Associated with each of the latter is an occupancy toggle for indicating whether or not that register contains an operand.

The action of the stack is defined in Table I. The following notation applies: S, A, B, T_A, T_B represent the contents of the stack counter , arithmetic registers, and occupancy toggles. X is an operand. ⊖ and ⊕ are unary and binary operators. S* signifies the contents of the cell addressed by the contents of S.

Now, while the case of stack operation, which is presented is somewhat simplified, it does show how a Polish string program can be mechanized. The occurrences of operators with $T_A = T_B = 0$ is actually abnormal, but are included to allow continuation in the event that the evaluation of an expression is interrupted. The state $T_A = 1$, $T_B = 0$ is unstable.

It is worth noting that upon completion of the evaluation of a well-formed expression we have $T_A = 0$, $T_B = 1$, with the value of S upon completion equal to its initial value. This offers a possibility for checking in the hardware which does not exist in such a simple form for the usual command language.

In the event that the reader does not happen to be familiar with the Lukasiewicz notation, he may find it useful to trace the operation of the stack for the program XY + VW/U + x Z + which corresponds to the expression (X + Y) x (U + V/W) + Z.

Subroutines and ALGOL Procedures

To realize additional important advantages from this program format, we extend these notions to handle n-ary operators or n-place functions that are defined by sub-programs. The important case of call-by-name is deferred. Call-by-value only is discussed. Declarations of functions will cause subroutines to be generated and extensions of the operator set to be defined. Similarly, it is assumed that for each array one or more storage-mapping functions have been defined and that, corresponding to each, a subroutine has been created from the array declaration which will have access to information on bounds of indices and the base address of the array. Such subroutines will also be called by extensions of the operator set. The program corresponding to the array element or function call will then consist of an ordered set of expressions representing the indices or arguments. The values of these are entered into the stack. When the operator corresponding to that subroutine is encountered, linkage automatically results.

Subroutine Control Using the Stack

It is now necessary to place the contents of registers A and B (or B if A is empty) into the stack since the subroutine to be executed will generally require these registers. Furthermore, the contents of the control counter, C, must be saved to enable return from the subroutine. This return can be saved in the stack and the position of this address recorded in another address-length register designated F. To afford a link to the return for a possible higher level subroutine, the former contents of F are retained in the cell with C. Finally, the subroutine entry address specified by the extension operator is entered into C and linkage to the subroutine is complete.

Within the subroutine, the parameters are addressed minus relative to F in reverse order. Temporary storage is allocated for the subroutine by advancing S corresponding to the number of cells needed. These cells can then be addressed plus relative to F. It will also be useful to store constants used by the subroutine ahead of its entry and address them minus relative to C.

Upon exit, the resulting value is left in B, the contents of S are replaced by F, and for the cell then addressed by S, the C-part will go to C and the F-part to F. Finally, S is reduced by n - 1. This automatic linkage construct enables the use of subroutines in depth and a subroutine may call itself.

Denoting the F and C parts of S* by S^*_F and S^*_C, respectively, the location of the subroutine by P, and other notation as previously defined, the action upon entry and exit is displayed in Table II.

Let it now be assumed that the execution of a scanned program element is delayed until the following syllable has been interpreted. Operators can be defined which force the preceding syllable to be a call-by-name. Each word in the stack has a control bit that distinguishes between values and names. The address referenced by the element preceding the call-by-name operator then is entered into the stack and the control bit for that cell set to indicate a name. If an operator is encountered, followed by the call-by-name operator, the operator (or the location of the subroutine which effects execution of an extended operator) goes into the stack and the control bit is set. Within a subroutine, any reference to this stack cell that is not followed by a call-by-name operator will cause execution. If, otherwise, the reference to the cell is again followed by a call-by-name operator, the cell contents are copied into the new stack level. A similar action results whenever a parameter of one subroutine is used in a lower level subroutine.

Other Consequences of the Design

Some of the key aspects of the logical organization of the machine have been introduced in a gradual fashion. While not all of the consequences of the model which have been developed can be presented in this paper, a few concluding remarks may be of value.

Change of sequence is accomplished by one of two means: jumps relative to C of conditional or unconditional type, or via a switching table of entries corresponding to labeled program segments. The conditional jumps examine the truth value in the stack produced by evaluation of a Boolean expression, and then cause it to be erased. For ease of segmentation and effective use of a random-access secondary storage device, we make program invariant with respect to its position in storage. Corresponding to each declaration of array, switch, or procedure will be a " locator " word which is assigned in a table. Program references are then made to these words to obtain the location of the corresponding program. These words contain other information which is itilized in multi-programming and automatic segmentation control.

"Locator" words also correspond to labeled program segments and input-output control information. The latter are grouped for scanning by a universal input-output control program which assigns I/O channels. The main program and I/O control program communicate via a status bit in these words.

Character and bit manipulation constructs for the machine are also departures from familiar practice, but arise from different considerations and will not be discussed here.

Conclusion

It is hoped that a case has been made for a way to introduce some new ideas into a field where enormous amounts of technical talent are spent in designing the hardware and programs for a large number of very similar machines. Most of these are "general purpose." Much of the effort in developing these machines could better be spent in designing some useful "special purpose" computers. An ALGOL machine would fit into the latter category. With automated logical design and fabrication in the immediate future, any number of these useful special purpose machines can be envisaged.

* * * *

This paper is based on work sponsored by the Burroughs Corporation.

References

1. Naur, P. (Editor): Report on the Algorithmic Language ALGOL 60, Comm. Assn. Comp. Mach. 3, No. 5 (1960), 299-314.

2. Burks, A.W., Warren, D.W., Wright, J. B.: An Analysis of a Logical Machine Using Parentheses-Free Notation, Math. Tables Aids Comp. 9 (1954), 53-57.

3. Bauer, F. L.: The Formula-Controlled Logical Computer "Stanislaus," Math. Comp. 14, No. 69 (1960), 64-67.

4. Samelson, K. and Bauer, F. L.: Sequential Formula Translation, Comm. Assn. Comp. Mach. 3, No. 2 (1960), 76-83.

	OPERAND	UNARY OPERATOR	BINARY OPERATOR
$T_A = T_B = 0$	1. $X \rightarrow A$, $T_A = 1$ 2. $A \rightarrow B$, $T_B = 1$, $T_A = 0$	1. $S^* \rightarrow B$, $T_B = 1$ 2. $S - 1 \rightarrow S$ 3. $B\ominus \rightarrow B$	1. $S^* \rightarrow B$, $T_B = 1$ 2. $S - 1 \rightarrow S$ 3. $B \rightarrow A$, $T_B = 0$, $T_A = 1$ 4. $S^* \rightarrow B$, $T_B = 1$ 5. $S - 1 \rightarrow S$ 6. $AB\oplus \rightarrow B$, $T_A = 0$
$T_A = 0$ $T_B = 1$	1. $X \rightarrow A$, $T_A = 1$	1. $B\ominus \rightarrow B$	3. $B \rightarrow A$, $T_B = 0$, $T_A = 1$ 4. $S^* \rightarrow B$, $T_B = 1$ 5. $S - 1 \rightarrow S$ 6. $AB\oplus \rightarrow B$, $T_A = 0$
$T_A = T_B = 1$	1. $S + 1 \rightarrow S$ 2. $B \rightarrow S^*$ 3. $A \rightarrow B$ 4. $X \rightarrow A$	1. $A\ominus \rightarrow A$	6. $AB\oplus \rightarrow B$, $T_A = 0$

TABLE I

	ENTRY	EXIT
$T_A = T_B = 0$	6. $S + 1 \rightarrow S$ 7. $F \rightarrow S^*_F$, $C \rightarrow S^*_C$ 8. $S \rightarrow F$ 9. $P \rightarrow C$	(Normal for procedure without value) 1. $F \rightarrow S$ 2. $S^*_F \rightarrow F$, $S^*_C \rightarrow C$ 3. $S - n - 1 \rightarrow S$
$T_A = 0$ $T_B = 1$	4. $S + 1 \rightarrow S$ 5. $B \rightarrow S^*$, $T_B = 0$ Steps 6 through 9	(Normal for procedure with value) (Same as above)
$T_A = T_B = 1$	1. $S + 1 \rightarrow S$ 2. $B \rightarrow S^*$ 3. $A \rightarrow B$, $T_A = 0$ 4. $S + 1 \rightarrow S$ 5. $B \rightarrow S^*$, $T_B = 0$ Steps 6 through 9	(Improper)

TABLE II

THE JOVIAL CHECKER
AN AUTOMATIC CHECKOUT SYSTEM
FOR HIGHER LEVEL LANGUAGE PROGRAMS

Mildred Wilkerson
System Development Corporation
Paramus, N. J.

Knowledge of specific machine language is still required for checkout of higher level programs. Consequently, several potential advantages of higher level programming languages have not materialized. These include shorter and less technical programming training, and faster program checkout.

JOVIAL is a higher level programming language. The "item" is its basic unit of data. The Checker is a program which executes translated JOVIAL programs and selectively records test results in JOVIAL language. Most non-essential information is eliminated from printouts. Check of actual results against expected results may be done automatically.

The Checker is part of a utility system which includes a Compiler and the Checker. The Compiler checks legality of JOVIAL statements, translates to binary code for a specific machine, and produces a JOVIAL program listing.

The Checker operates the object program and "snaps" every modified value of selected items. Basis of selection may be items of interest to the programmer, items whose final values deviate from programmer-supplied, expected values, or both. Each modified item value is printed with the statement which modified its value at that point and an associated label. Final values only of any or all tables may also be recorded.

Introduction

Although higher level languages have been in use scarcely two years, the contributions of FORTRAN, COBOL, JOVIAL, ALTAC and other such languages are now beginning to be realized by their users. Despite continuing machine obsolescence, the problem of program obsolescence due to the differing languages of various computers is now soluble with higher level language programs and compilers.

Most programmers observe that, while overall output may not yet reflect the speed-up, coding with a higher level language is considerably faster than coding in machine language. More time is now available for such problematic areas as problem analysis, program design, and modifications resulting from system design changes.

The pitfalls leading to trivial, clerical type coding errors also have been radically reduced by higher level language. This is generally attributed to the sheer reduction in the number of higher level language instructions required to program a given problem, to the more flexible format of instructions, and to the greater readibility of the language.

On the other hand, one major, potential advantage of higher level languages has not been realized. This is the elimination of machine language as a requisite.

The Problem and Its Consequences

Higher level languages have not yet replaced machine languages for a single programmer. The expectation in this respect was that machine languages could be omitted from the training of all programmers except those involved in programming and maintaining compilers and special utility programs. The existing situation, however, is that programmers must still be taught the symbolic language of at least one specific machine, as well as the new higher level language. Consequently programming training today is more time-consuming and more complicated than ever before.

Closely associated was the hope that higher level languages would further progress toward a common communication link between man and machines, between non-specialists and computer specialists, between management and programmers. One of the deterrents toward this goal is that while such a language may exist, conventional training is still dominated by machine languages. This training is neither appealing nor expedient for management and non-specialists. It is time-consuming and tedious. It requires a capacity for rote memorization and a fastidiousness for clerical detail. The very technical nature of machine language furthermore restricts the type of people who may be selected for programmer training, and may, in fact, discourage many creative people from entering this profession.

It is not resistance to higher level languages which has prevented relinquishment of machine languages from programmer-training. It is the fact that no method had been devised to produce test results from higher level language programs that did not depend upon a thorough knowledge of machine language by individual programmers.

The JOVIAL Checker is designed to solve this technical problem. Its consequences, or any similar solution, cannot be displayed as a cure for the problems just mentioned. It can, however, be regarded as one stepping stone toward full realization of the advantages of higher level languages. To our knowledge, it is the first utility program designed to eliminate the last remaining traces of machine language from higher level language programming.

The Problems at SDC

The problems that gave rise to the Checker were much more modest. Program checkout has long been one of the most time-consuming headaches of the programming field. While higher level languages have significantly reduced the number of errors made in the original program, virtually no time has been saved in debugging and checking out programs.

Professor Wrubel[1] cited the problem in apt words when discussing one of the better known higher languages. "It is a frustrating thing," he writes, 'to have computed the answers correctly only to find them printed on the output page in an all but incomprehensible jumble."

A survey of the practices within the field yielded no satisfactory solution. Generally, test results are printed in the machine language format. Instructions are sometimes traced, but this produces stacks of printout paper with no clue to the origins of errors. An individual programmer with a great deal of foresight may leave holes in his program for insertion of instructions that could produce test results in any format, provided these were prepared by the programmers and later deleted from his program.

Due to the efforts of Jules Schwartz and others at System Development Corporation, we have been programming in a higher level language called JOVIAL for well over a year. Various utility programs for program checkout have been developed, but none embodied all of the needed capabilities. Martin Blauer, therefore, initiated the efforts that led to the design of the Checker to meet the following requirements:

1. All test results appearing on the printout, including instructions, data or other information, should appear in the JOVIAL format.
2. Adequate information should be provided to locate the origin of errors, but otherwise unneeded or unwanted results should be omitted from the printed test results.

The JOVIAL Checker

The JOVIAL Checker was designed and developed by members of System Development Corporation for use in checking out programs written in the JOVIAL language. It will be available for use in March 1961, and will operate upon programs translated by the JOVIAL-to-IBM 7090 Compiler. The Checker is also suitable for use with the AN/FSQ-31V military computer and is readily adaptable for use on any other computer for which a JOVIAL compiler is available.

The combined Compiler and Checker system is designed to translate programs written in JOVIAL language, execute the translated instructions, and produce test results in JOVIAL format. A brief examination of the organization of JOVIAL variables and one type of JOVIAL instruction, as well as an outline of the Compiler's functions, will be helpful in understanding the Checker's operations and output.

Organization of Variables

All input and output data, as well as variables manipulated internally by a JOVIAL program are organized into items, entries and tables. The item is the basic unit of data. Its size may range from one bit to the total number of bits in the machine word, depending upon the size of the data it will contain.

An entry is comprised of one or more items. Two or more items collected in the entry are usually related, i.e., a payroll code, an employee number, a tax deduction rate, etc. relate to one employee.

A table is comprised of one or more entries, usually repeating the same group of items contained in the first entry. Each entry, however, contains a different set of variables. There is no practical limit to the size of either an entry or a table.

All items and tables used by a JOVIAL program are defined according to the basic characteristics of the data they will contain and are assigned symbolic names. Thereafter they are conveniently called upon by name. Entries are referenced by integer values in the form of constants, subscripts or other variables.

Assignment Statement

Dynamic JOVIAL statements may be classified according to two basic types--those which control sequence of operations and those which modify the language may be used to modify the value of any variable--the assignment statement and the exchange statement. The latter is a type of two-way, restricted assignment statement.

The assignment statement places the value named by the right term into the location of the variable named by the left term, altering the format of the right term to fit, if necessary.

```
example: (Left Term)  (Right Term)
           NUMBER   =     1 $
            ABLE    =  ABLE + BAKER$
```

The item named NUMBER in the first example is assigned the decimal value of 1. In example two, item ABLE is set to its own value plus the value of item BAKER.

Since modification of the value of any JOVIAL variable must be performed with this type of statement, the assignment statement, and particularly its left term, is vital to the operation of the Checker.

The Compiler

The JOVIAL Compiler accepts a program written in JOVIAL, analyzes its statements for illegalities, generates an intermediate language version of this program and then translates from this language to the binary code of the specific machine. Four of the Compiler's working tables are saved for subsequent use by the Checker. One relates to statement labels; one gives references to items and tables; another, to so-called status items; and the last refers to intermediate language statements.

The output from the Compiler is the object program and test data, the above tables and a printer destined tape, used also by the Checker, listing each JOVIAL statement with its equivalent symbolic instructions and octal machine code. The listing also provides a record of input test data in JOVIAL format and, if any illegalities were detected, error messages in context. When the JOVIAL program is corrected of all errors, it is ready to be run with the Checker.

The Checker Options

The general functions of the Checker are to operate the object program with the supplied test data and to record selective test results in JOVIAL format on a printer-destined tape. Results may also be printed on-line, if desired.

To use the Checker, the programmer creates two or three control cards to specify the method of selecting test results for recording. From three basic options of selecting test results, the programmer may choose any, all, or none. By answering 'yes' or "no" to each of the following questions, the programmer has eight combinations of recorded results from which to choose:

Unconditional Trace: Are dynamic snaps of selected items for which the programmer has not supplied expected final values wanted? (Let us call this an "unconditional trace." Dynamic snap, as opposed to final or static snap, is used here to mean that every value of a selected item is recorded each time it is modified throughout the entire operation of the program.

This option also applies to items in selected entries. If, for example, the program operates upon every other entry containing the selected item, instead of every entry, the programmer has no need for any dynamic snaps of item values located in half of the entries. The programmer then specifies the item name followed by the entry number. This selectiveness is important because the total number of items selected for an unconditional trace is limited to one-hundred items, and each iteration of the same item in different entries is counted as one item.

Strings and items located in tables whose entry lengths or entry structures vary from entry to entry may also be traced.

To initiate this type of trace, the programmer creates an unconditional trace control card, followed by a sufficient number of cards to list every item he wants traced in this manner. (See figures 1 and 2).

As a result of the unconditional trace, the values of all modifications of selected items are recorded for printout. Each value is accompanied by the item name and entry number, the assignment statement which modified the item at that point, and the closest preceding JOVIAL statement label.

Discrepancy Trace: Are dynamic snaps wanted only in the event that final values of items within selected tables deviated from expected values? (Let us call this a "discrepancy trace.")

For the discrepancy trace, the programmer supplies a control card with the words, "Discrepancy trace," and defines two sets of tables as part of his organization of variables with the original JOVIAL program. One set of tables defines and names all items selected for dynamic snaps in the event their final values are in error. These tables are named "ACTØ, ATC1," etc. After the object program has been operated, the actual final values of the items defined within these tables automatically will be placed in the item's assigned location.

The second set of tables are given the names, "EXPØ, EXP1," etc., and contains the programmer-supplied expected final values for all items named in the ACT tables. The values of items within the ACT tables must correspond exactly with the positioning of expected values in the EXP tables, and all recurrences of the selected items in every entry of the tables must be provided for.

After operation of the object program, actual final values of all items in the ACT tables are compared with the expected values in the EXP tables. Only in the event that a discrepancy occurs between any of the correspondingly positioned values, is a trace initiated.

Except that only those items in error are traced, this trace is performed in the same way as an unconditional trace, and recordings will also be accompanied by the modifying assignment statement and closest preceding statement label. Although any number of items may be placed in the ACT-EXP tables, only the first one-hundred discrepant final values will be traced.

When an item to be checked is already organized within an entry containing different items which need not be checked, the programmer may remove the selected item from its original table and define it within an ACT table. This reorganization in no way alters the operation of the program or the results obtained.

Final Snaps: Are final values only of selected tables wanted? This option will usually be employed in conjunction with one or both of the options already discussed. In effect, it is a 'static snap" of the values in preselected tables at the end of the program's operation, therefore no assignment statements or labels accompany these recordings. Table names, entry numbers and item names are provided. On the control card the programmer may specify that final snaps be made of all tables defined with his program, no tables, only the tables named, or all tables excluding those named. (Figure 3)

With any of the three options named, special information must be supplied to the Checker if recordings are requested from tables whose entry lengths or entry structures vary from entry to entry. One control card is needed, one card for each table name, and one or more cards per item. Control items--or those items containing information about the length or structure of each entry --are designated by their absence or presence in floating fields. Control information on strings is also specified on these cards.

Highlights of Checker Operation

Three major routines called Control, Tracer, and Record constitute the Checker program. These routines, in turn, are modularly constructed of multiple subroutines for both flexibility of operation and ease of modification. The broad flow of operations is illustrated in Figure 4 to depict the sequence of the operational highlights only.

One pass is required through the Checker program unless actual values deviate from expected values, in which case two passes are made. The Checker may be operated in a strictly unconditional mode, strictly discrepant mode, or a combined mode. If both modes are desired, an unconditional trace control card is used, but the first pass succeeds in performing all the operations required of the first pass of both modes.

For an unconditional trace, selected item names are read in from card reader or from script tape. These item names and entry numbers, if entry references are furnished, are entered into a table called "Trace."

Statement References: With the aid of tables furnished by the Compiler, the items named in table Trace are then used to locate all JOVIAL assignment statements which contain these items as their left terms. As these assignment statements are located, they are placed in a table called "Refer." In addition to all JOVIAL statements which modify the items selected for trace, the Refer table contains the relative location in the binary program of the machine instruction which modifies the value of the item. This instruction is usually a "store" class instruction. The Refer table is subsequently used to insert traps in the object program, create a table consisting of displaced "store" instructions, and finally, its assignment statements are recorded for printout with corresponding item values.

Statement Labels: Two compiler tables are used to associate the closest preceding JOVIAL statement label with each assignment statement in the Refer Table. A search of the intermediate language table yields only the operator "label" and a reference to another table containing all statement labels. JOVIAL labels are easily recognized, however, and these are saved until associated with an assignment statement to be traced or until another JOVIAL statement label is encountered. The last label saved is thus automatically associated with the next assignment statement under trace.

Imbedding Traps: So that all modifications of an item under trace may be saved before the item is subjected to further modification, the object program is imbedded with "traps." Traps may be defined as instructions which effect an unconditional transfer of control to the snap recording routine.

Traps are imbedded in the object program to replace each "store" class instruction whose relative location is furnished by the Refer table. The store instructions, in turn, are relocated in another table and are operated upon from within this table prior to operation of the Snap routine.

Snap Tables: Recordings of the modifications of all items under trace are saved in a snap table which has a capacity of 400 snaps per Checker pass. If snaps exceed this capacity, the contents of the filled snap table are repeatedly buffered onto a scratch tape and brought back into memory just before the final recordings for printout are made.

Operation and Recording: The Checker then operates the object program with imbedded traps. If control information is present regarding variable length or variable structure entry tables, this is tabulated. Final snaps of selected tables are processed as requested and recorded on the printout tape. If no ACT tables are present, snaps resulting from the unconditional trace are grouped with appropriate statement labels and assignment statements. These are recorded on the output tape and the job is logged complete.

Discrepancies: If ACT tables are present and one or more values deviate from the EXP values, the Trace table is recreated. This time, however, instead of containing items requested for an unconditional trace, the Trace table contains only the names of items which revealed discrepancies. Again, traps are imbedded, and again, the program is operated. For the second pass, all parts of the Record routine are omitted except the final recording of snaps and associated information destined for printout.

Checker Printout

The printout of the Checker provides the programmer with the following information in the JOVIAL format: (Figure 5)

1. Final values of any tables specified for final snaps. The word "Table" precedes the table name, the word "Item," its name, 'String," its name, etc. Fixed entry length tables are printed first, listing all values for each item sequentially. Variable structure or variable entry length tables follow, however, these values are listed entry by entry because the same items may not be present in all entries.
2. Unconditional traces are so labeled and are followed by all such traces in the format previously described.
3. Discrepancy traces are listed last in the same format.

Checker Restrictions

Restrictions imposed upon JOVIAL programs by the Checker at this time are approximate since the Checker, which is itself a JOVIAL program, has not been compiled at the time of this writing. Restrictions estimated for the Checker and the JOVIAL program when compiled on one computer will not be the same for another computer with greater capabilities.

When compiled on the IBM 7090, the Checker is expected to occupy about 16,000 registers, reserving 10,000 for the object program and test data. Most of the remaining core space will probably be occupied by the Compiler tables and the tables created by the Checker. Actually no core space is left unused, thanks to one whimsical programmer who filled in remaining space with transfer to a routine which prints the message, "Dear Programmer. You transferred out of your program. How about that?"

Practical limits to the lengths of tables created internally by the Checker impose two rather generous limits on the JOVIAL program. It is assumed that the number of assignment statements which refer to any item under trace will not exceed an average of four. JOVIAL assignment statements are also limited in length to an average of six registers each.

Evaluation

In light of the stated objectives of the Checker, comments must be withheld until programmer use of the utility program is observed and indications of time-savings are available. Similarly, the overall efficiency of the Checker is more convincingly reported after compilation figures and timing results are tabulated.

If, however, a bad plan admits of no modification, with a slight twist of logic we can believe the Checker is a good plan in at least one respect. Two proposals are now under investigation for possible modification of this or future Checkers.

One proposal considers that the programmer may wish to check out his program several times with different sets of test values, all of which may not necessarily be included with the original program. Once the program has been compiled and corrected, the present system necessitates recompiling to include such changes. The modification under consideration would enable the programmer to insert, delete, or change values of items by means of an additional Checker sub-routine which would operate at the beginning of the Control routine.

The second proposal permits the programmer to designate certain items as "conditioning items." Conditioning items are those items not selected for a trace, but whose use in the program affects the values of items selected for a discrepancy trace. Conditioning items would be traced unconditionally in the event that the item with which they were associated were found to be in error.

The present system accommodates such items only as items selected for an unconditional trace. Should the main item selected for a discrepancy trace not be in error, unneeded results would be produced.

Reference

1. Wrubel, Marshall H., A Primer of Programming for Digital Computers, McGraw-Hill Book Company, Inc., N.Y., N.Y., 1959, p. 122.

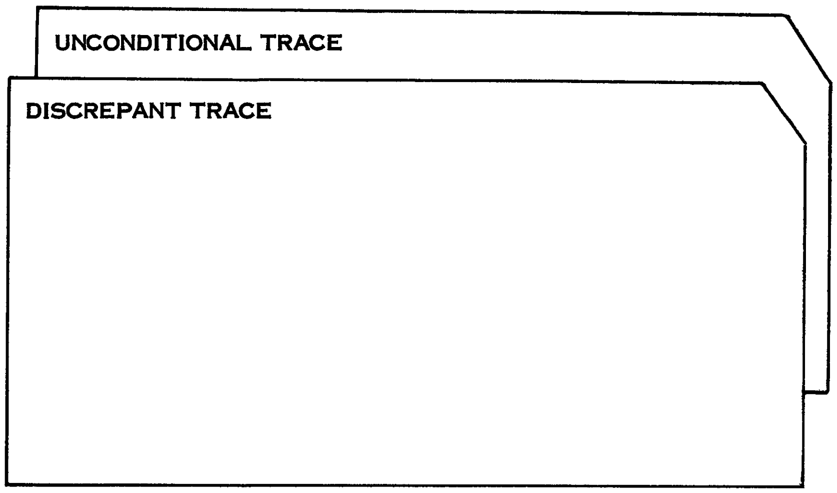


FIGURE I TRACE CONTROL CARDS

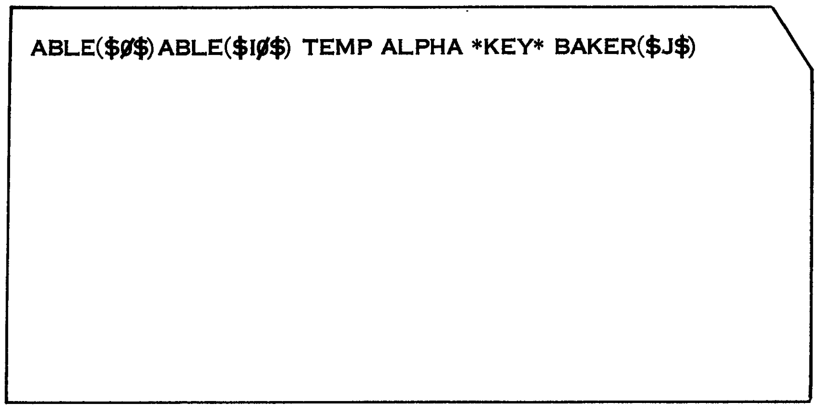


FIGURE 2 AN ITEM CARD FOR UNCONDITIONAL TRACE

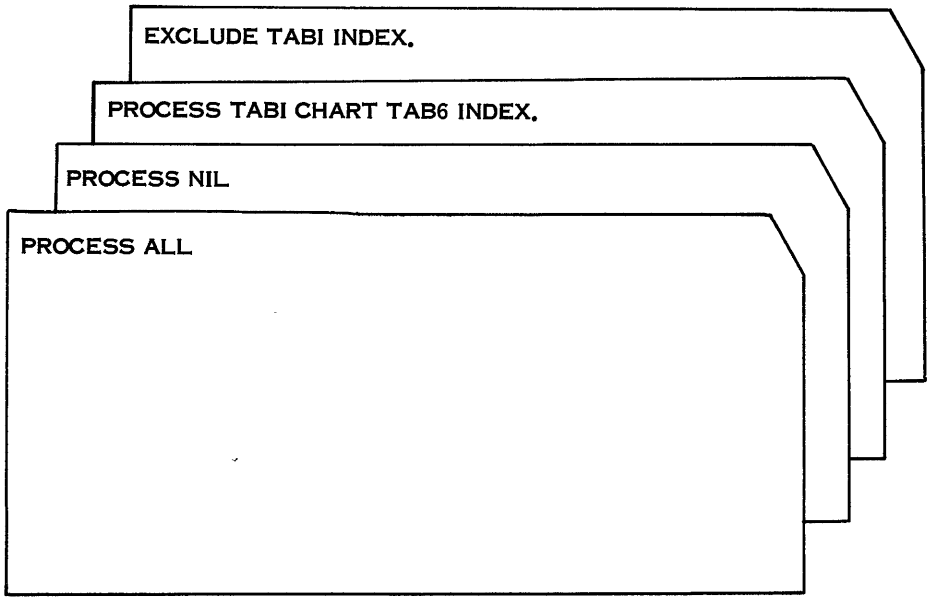


FIGURE 3 FINAL SNAP CONTROL CARDS

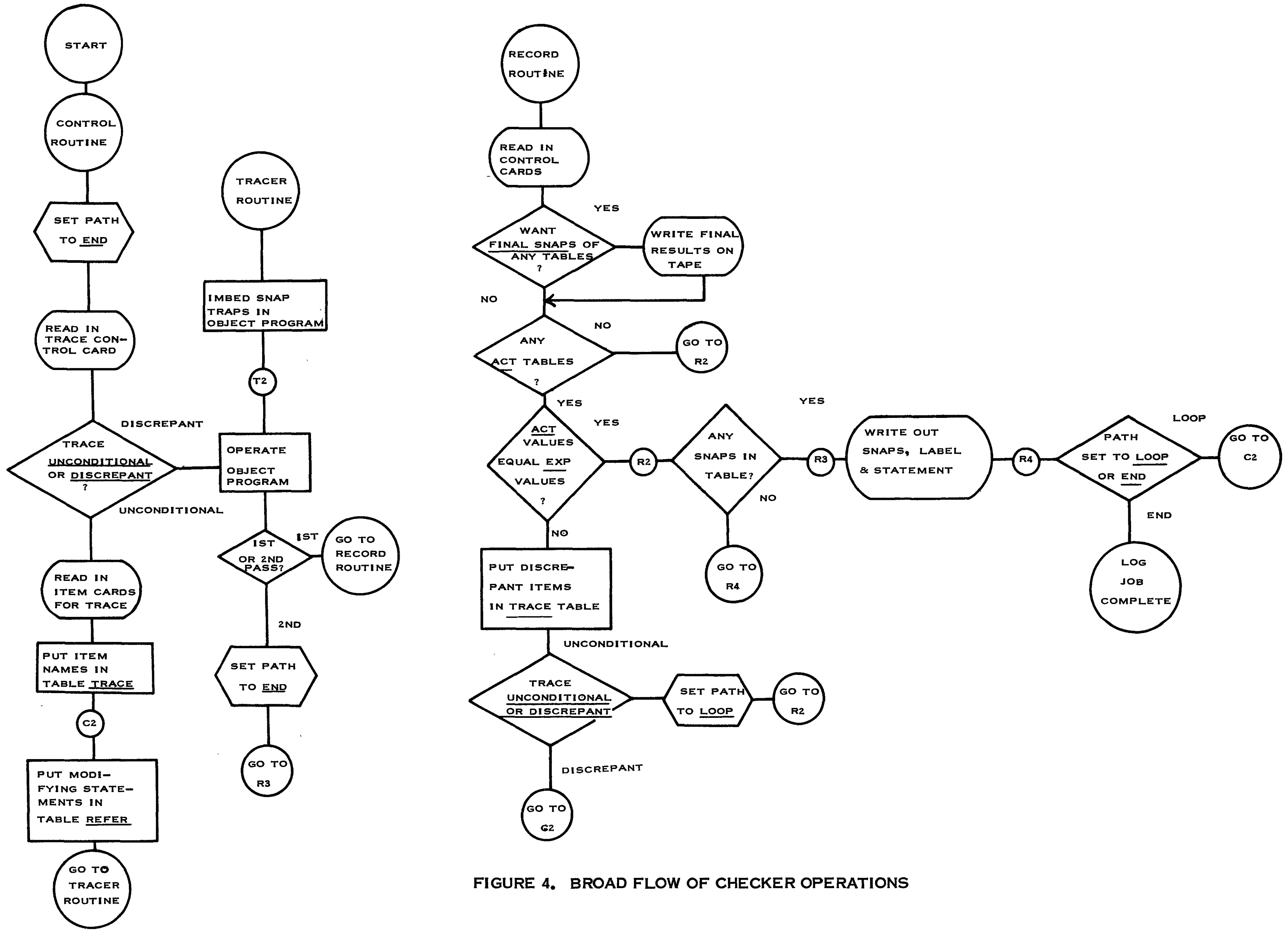


FIGURE 4. BROAD FLOW OF CHECKER OPERATIONS

FIGURE 5. SAMPLE CHECKER PRINTOUT

```
TABLE DECOR
    ITEM COLORS
    ENTRY   (0) BLUE        (1) GREEN       (2) ORANGE
            (3) BLACK       (4) PURPLE      (5) PINK
            (6) RED         (7) YELLOW
    ITEM PATTERN
    ENTRY   (0) .135663E3      (1) .06025E2      (2) .2E1
            (3) .55E3          (4) .200E3        (5) .5E-2
            (6) .601E3         (7) .325E0
    ITEM NAMEP
    ENTRY   (0) BLUBEL         (1) SPRING        (2) SUNSET
            (3) EBONY          (4) DUSK          (5) SUNRIS
            (6) BLAZE          (7) JONQUL
```

FINAL VALUES IN TABLE WITH FIXED LENGTH ENTRIES.

```
TABLE MILLS
    ENTRY   (0)
       ITEM PATTNO        .135E3
       ITEM TLMILS        ONE
       STRING LOCATN  SAVANA
    ENTRY   (1)
       ITEM PATTNO        .2E1
       ITEM TLMILS        NONE
    ENTRY   (2)
       ITEM PATTNO        .6025E2
       ITEM TLMILS        THREE
       STRING LOCATN          LOUSVL
                              SAVANA
                              BIRMHM
```

FINAL VALUES IN TABLE WITH VARIABLE LENGTH AND VARIABLE STRUCTURE ENTRIES.

```
UNCONDITIONAL TRACES FOLLOW

    10A. BYTE ($B, E$) (PRICE ($C$))       TEMP ($A$)
       (2) (3) (PRICE (12))       .99
    10A. BYTE ($B. E$) (PRICE ($C$))       TEMP ($A$)
       (2) (3) (PRICE (13))       .87
    10A. BYTE ($E, E$) (PRICE ($C$))       TEMP ($A$)
       (2) (3) (PRICE (14))       .54
```

DYNAMIC SNAPS OF VALUES OF PARTS OF ITEM PRICE.

```
DISCREPANCY TRACES FOLLOW
    CF8.  ABLE ($L$) COUNT $
          ABLE (0)     2
    CF8.  ABLE ($L$) COUNT $
          ABLE (1)     3
    CF8.  ABLE ($L$) COUNT $
          ABLE (2)     4
```

DYNAMIC SNAPS OF VALUES OF ITEM ABLE

FACTORS AFFECTING THE CHOICE OF MEMORY

Claude F. King
Logicon Inc.
Palos Verdes Estates, Calif.

Introduction

One of the fundamental choices in the design of a computer or data processing system is concerned with the medium for the storage of information. Once the system requirements impose the need for information storage that exceeds a certain level, typically several hundred bits, a "hierarchy" principle of memory[1] sets in. As expressed by Von Neumann, the system may call for an overall storage of N words at an access time t. For economic or other reasons it may be more practical to provide for some smaller quantity of storage N_1 with an access time $t_1 = t$ and obtain the total storage N at some longer access time depending on the system needs. Extending this reasoning then to the more general case there would be a sequence of capacities $N_1 < N_2 < . . . < N_{k-1} \leq N_k$ with access times $t_1 < t_2 < . . . < t_{k-1} \leq t_k$ so that N_i words are required for each access time t_i. Then each value of i would represent one level in the hierarchy of memories, and the hierarchy has k such levels. In many modern computers there are also quasi-levels that don't fall into Von Neumann's ordered sequence of levels. Certain bodies of the information storage may have special characteristics or requirements that call for different storage media that would yield greater capacities at shorter access times or vice versa that may not fall strictly into the ordered sequence. In making the choice of memories for a given system the characteristics for each level should be considered. The starting point should be the set of system requirements. These would be examined to determine the different levels that may be accommodated to satisfy the requirements. A set of requirements would then be generated for each level. An initial choice might then be made for each level. Since the choice of one level could strongly influence another they should be examined together and appropriate changes made. It should be borne in mind that several levels of the hierarchy may exist within one medium, for example the complete tracks as compared to the short recirculating loops on a magnetic drum would represent different levels. References to specific types of memories in this discussion such as core or drum will usually be in the generic sense. For example, the family of core memories includes twistors, thin films and multi-aperture cores, as well as the conventional torroids. Also, the comments on drum memories will usually apply to the disc, as well as the cylindrical configuration.

The Determination Of The Hierarchical Levels From The System Requirements

For certain applications, such as the use of the computer in a control system for a well-defined purpose, the procedure can be straightforward. The computer's task can be described by a set of equations along with the required computation rates. The equation set should be separated into each of its modes and the needed capacity and access times evaluated separately. For each mode the information should be divided into different kinds that may impose different requirements on the storage. For example, the intermediate results of computations have different requirements from the storage of the instructions. Considerations should be made at each level as to the need for special protection for the information storage. For example, can volatility be tolerated for the section of the storage under consideration. The need and frequency of information change should be taken into account for each level. Some of the storage levels may be able to tolerate different degrees of unreliability than others. The tolerance to environmental conditions may be more critical for some levels than others. Where power consumption may pose problems it may be possible to operate at different power levels, depending on the mode. When each of the levels has been defined by these considerations, the possible choices for each level can then be made.

For many applications the computer's use is multi-purpose and considerations must be made on the basis of worst-case expected problems and highest-use types. Different types of problems can be looked on as different modes of the system. The flexibility to handle many different kinds of tasks is an overriding consideration itself, that often dictates maxima in both speed and capacity limited only by economic considerations. This type of application leads to the ordered type of level identification that is based primarily upon access time and capacity.

Sample Set of Levels

As an example of a set of levels that might be derived from a hypothetical computer system, consider the following:

1. Level # 1 consists of the active storage elements of a system that would be changed at computer clock rate. These are the single bit elements that take part in the computer control, arithmetic, input and output buffering and buffering into other levels of storage. In most modern computers they are transistor flip-flops, and for most controls computers the total capacity of this kind of storage runs from 50 to 500 bits depending upon how the machine is organized and the task it performs. Access times for this level of storage is usually from 1 to 10 microseconds. Since this level of storage is usually the most costly in terms of numbers of components per bit, an attempt is usually made to restrict the number of bits of this type to as small a number as possible, perhaps at the expense of some of the other levels. Loss of memory with removal of power is usually not a consideration at this level, although occasionally it is protected by providing for a complete dump into another level on the sensing of a power drop-out. Protection against loss of memory from a power drop-out can also be obtained by the use of magnetic cores either alone or in conjunction with flip-flops arranged to hold a state on sensing a power drop-out. Since at this level the storage elements communicate directly with the logic elements and through the logic elements to other storage elements they must have a power amplification greater than unity which is not generally true for other storage levels.

2. Level # 2 might consist of from 100 to 1000 bits with a word access rate of from 10 to 100 microseconds. This storage may be provided for special input or output functions where incremental techniques could be applied at rates beyond that necessary for other portions of the computations. Such things as delay lines, magnetic core stepping registers and drum recirculating registers are commonly applied for this storage level.

3. Level # 3 could contain the registers that take part in the arithmetic operations. In most configurations this is not a separate identifiable level, the functions being provided by either the level # 1 or level # 2 storage or part by both. However, it has its own characteristics that could lead to a choice of a separate media. Its access time requirements may be longer than that required for levels # 1 and # 2.

4. Level # 4 in this set is taken to be that section in which the intermediate results of computations are stored. The access time requirements for this level are probably the same as for level # 3 and in some cases levels # 4 and #3 are combined. The capacity of this level usually varies between 100 and 1000 words. In some computers where levels # 2 and #3 described here are combined with level #1 the intermediate results store represents the first level in which a bulk type store is utilized.

5. Level # 5 in the scheme described here is the first level in which the instructions are stored. For this level one might conceive of a group of high-speed, wired-in subroutines in which a certain speed advantage may be derived at a minimum cost.

6. Level # 6 might be a memory that would be combined with level # 4 in which a large number of instructions are stored in a manner that is readily changed. This is the work horse of the large data processing computers and contains up to 100,000 words in some. Access times vary from a millisecond to a microsecond in present computers.

7. Level # 7 could be another wired-in type of storage that could be varied by a plug-board arrangement or some other means to provide for a body of instructions that are protected against memory loss due to electrical phenomena yet may be changed in a reasonable (several days) span of time. For a drum memory there might be some instructions with write amplifiers active that would place them in level # 6. Level # 7 could be associated with tracks in which the write amplifiers are deactivated or no longer present after recording. The distinction between the two is sometimes referred to as "hot" storage for level # 6 and "cold" storage for level #7.

8. Level # 8 would be a bulk storage of instructions that would block-transfer routines into the working store on program command. This level would probably be provided by magnetic tape or drum and need not communicate directly with the outside world.

9. Level # 9 would be similar to and perhaps combined with level # 8 and would contain the bulk store of data. It is listed separately from # 8 since it could have different access time requirements than # 8.

10. Level # 10 for this example is that body of storage associated with the "outside world" and may actually be a multiplicity of levels consisting of punched tape or cards or magnetic tape, or even printed pages.

The levels listed here are those that may be suggested by a particular set of requirements. It is expedient in the actual execution of the design to combine many of these, but in almost all computers at least three levels are easily identified.

Example

In order to understand more fully the implications of this discussion consider an application that would present an environment so harsh and unpredictable and require a reliability so great that the designer may be forced to carefully consider his choice of memory at each level in order to have any chance of success. For this example consider a computer used to control a spacecraft on a journey to a near planet. The difficulty of the design for this application arises from the need to obtain useful operation during and after the long span of time in transit. This could be from six months to a year to the nearest planets. In order to keep it as simple as practical we won't burden it with the tasks involved in the launch guidance, but it shall be the primary center of control and guidance functions from the point of final stage vehicle separation onwards. The functions it would be called upon to perform might include system sequencing, system diagnostic checking, command decoding and interpretation, telemetry sequencing and formatting, antenna pointing, star tracking, engine and control jet commanding, terminal guidance, payload data buffering, instrument calibration, environmental control, and system decision-making (to select alternate system modes in the event of a recognized subsystem failure). The problem of operation without maintenance for a period as long as one year is a severe one when looked at in terms of obtaining a probability of a successful mission as high as 90%. This implies a mean time to failure of 10 years for the system. One approach to this problem involves a computer designed to periodically diagnose itself, as well as the rest of the system, and take alternate actions by decisions itself or by supplying the diagnosis data by telemetry to earth where a decision can be made and a command issued back to the vehicle via the computer to alter the system, or computer itself, on the discovery of a malfunction. Consider now what this problem suggests in terms of different levels of memory.

1. There should be one section of instruction storage that contains the overall executive routine that must be wired in and designed in the most reliable manner possible. Brute force redundancy may be applied here and the section should be kept reasonably small.

2. In order to provide for the required operational programs plus diagnostic programs plus alternate routines and subroutines, the amount of non-working instruction storage required will be quite large.

3. The number of instructions working at a given time may be kept small so that the variable instruction store need not be large.

4. There may be a need for the storage of a large amount of payload data for a slow transmission to earth at a convenient time.

5. In order to keep the speed requirements for the rest of the computer and memories to a sufficiently low value to conserve power and assure good reliability a special input-output buffer memory may be needed to cope with the high rate data from the terminal guidance sensors.

6. The intermediate results memory may be called upon to function at a greater speed than the working instruction store and thus might be accomplished by a memory of slightly different design.

Considering the logic portion of the computer as another level and the storage buffer on the earth for special instructions that may be sent over the command link we have described a system in which eight levels of memory may be a reasonable solution. Besides the extreme reliability requirement, the premium for size, weight and power consumption will be such that each level should be examined in detail and the design tailored to the requirements of that level and the communication with the other levels. Redundancy of different kinds might be applied at different levels. For example, for the bulk stores, present missile and satellite magnetic tape recorders capable of millions of bits each are available at weights of only 5 pounds each making complete unit redundancy attractive there. For the intermediate results and working stores a capacity capable of the most complex task may be sacrificed in the event of detected malfunction by programming around the trouble spots with simple programs constructed and entered from the earth via the command link.

General Comments On The Choice Between Cores and Drums

The trend in memories for the large data processing computers, where a maximum in speed, power, and flexibility is desired, has been to magnetic cores for the working storage supplemented by magnetic tape units for bulk storage. For applications in which economy has been paramount, magnetic drums and discs are the most prominent types of memories in use today. For computers used in the control of military systems the choice is not as obvious and the question at times is controversial. Let us look at this one in a little detail. Some of the general characteristics of this application are as follows:

1. There is a desire to protect a certain critical body of the storage against loss of information due to acts of man or nature over an extended period and yet have it ready at instant's notice. This set of instructions and constants are usually well-defined.

2. Because of the inherent power and flexibility of the computer it is desired to use it for other less critical and less well-defined tasks to obtain overall system simplification.

3. The desired reliability and mean time between maintenance is usually beyond that easily obtained.

These kinds of requirements have generally led designers to choose a drum, and this is partly because we tend to be purists when it comes time to choose the store. If we must choose just one kind, the drum is a reasonable choice. It satisfies the desire for permanence of storage of requirement # 1, since we can store the information and deactivate the write amplifiers and still feel quite secure about its staying there. It satisfies the flexibility desired in # 2 since we have the ability to readily change instructions and constants by electrical means. The drum has generally fallen short of the desire expressed in # 3 primarily because the memory is dependent on mechanical motion and is susceptible to wear and a reduction in overall computer margin due to timing shifts that arise from the effects of a rugged environment. While this has been true it still stacked-up favorably on # 3 against a pure wired-in approach which sacrifices # 2, or a complete variable approach which gives up # 1 because of a 3 to 1 ratio in numbers of components that usually occurred in the implementation. This is a result of the serial nature of the storage, the use of time selection afforded by the rotation and the use of recirculators for arithmetic registers, fast access storage and input-output special buffers and processors. However, the tide is beginning to turn. Applying the multiple level principles expressed earlier we can take care of # 1 by a wired core arrangement embodying magnetic switch selection to hold the electronic components down. This can be arranged such that programmability is not completely sacrificed, requiring a two-day operation to perform, which is not at all unreasonable when looked at in terms of the time it takes to verify a good program when a change has been made. We can retain the features of # 2 by supplying another section of random access core memory, again employing magnetic switch selection to hold the electronic components down. This memory would serve as intermediate results memory when in the operational mode and also would be used for system service routines if given the ability to be addressed as instructions. These would be entered and operated piecemeal to keep the size of this section down. A third section of special input-output buffer of either core or magnetostrictive delay line would be added. To further make use of the speeds available with modern techniques a little application of microprogramming could drive the component count down still further. These things taken together will not still give us as small a component count as the drum, but brings the count perhaps within 2 to 1. With the best selection and utilization of modern electronic components this 2 to 1 ratio in component numbers in favor of the drum will not counterbalance the unfavorable mechanical aspects, and unless there is a significant and fundamental change in the way drums are constructed we can expect an ever-increasing percentage of military control computers with non-rotating memories.in the future.

Conclusion

There are many techniques and media for storage of information described at this conference. Each should be considered in terms of its special qualities and characteristics. Be not quick to rule out one in favor of another for there may be one level in a given application that may be specifically tailored to the media discarded in light of requirements for another. As computer designers become more sophisticated there will be a greater tendency to multi-level memories and a wide application for the many techniques.

1. John Von Neumann, "The Computer and the Brain", Yale University Press

A NONDESTRUCTIVE READOUT FILM MEMORY

by

Richard J. Petschauer
Rodney D. Turnquist

Remington Rand Univac®
Division of Sperry Rand Corporation
St. Paul, Minnesota

Abstract

A film memory system, utilizing BICORE memory elements and capable of nondestructive readout operation, was designed and built. Two thin films are used for each memory element: one of high-coercivity cobalt-iron alloy, one of low-coercivity nickel-iron alloy. The former is used to store the bit while the latter senses it. These films were vacuum deposited in a multilayer fashion onto a single glass substrate. The film cores are circular discs with a diameter of 0.050 in. and placed on 0.100-in. centers and fabricated in 16 x 18 arrays. A model memory containing sixteen 18-bit words was designed and constructed to evaluate the performance of this type of memory and to provide a device which could be used to rapidly test film core arrays. The design and operating characteristics of the memory are reported. Read currents can vary over a 2:1 range without producing errors. Temperatures and other environmental factors did not appreciably affect memory performance. Later, a 1024-word 36-bit memory using the same principle of operation was constructed and operated. Some of the design details as well as the preliminary results of this larger memory are reported.

Introduction

Considerable attention has been given during the past several years, to thin magnetic films as computer storage elements.[1,2,3,4] Most work relating to this subject was concerned with utilizing the fast switching property of films for destructive-readout (DRO) memories. Oakland and Rossing[5], however, described a method of achieving nondestructive-readout (NDRO) by use of two magnetically coupled film spots for each bit — one to store information and the other to sense it. The present article describes the design and operating characteristics of such a memory. It is termed BICORE memory because two distinct thin films make up each film core.

In general, there are two reasons why a memory should be capable of NDRO operation. They are as follows:

1) To eliminate alterations of stored programs or critical constants caused by transient readout errors
2) To permit higher operating speed by eliminating the necessity of re-writing after each readout.

The operating mode (for a BICORE memory elementary) described in this paper was chosen to maximize memory reliability and to remove need for exceedingly stringent control of film-core parameter. While speed was not considered of paramount importance, the memories described herein were operated at a 1.5 μsec cycle time. Other measurements indicate that other modes of readout can result in cycle times as short as 0.1 μsec.

The objectives of the described effort were 1) to construct a small memory and then to test its design in an electrical environment similar to that of a complete computer, 2) to produce a device for rapidly testing complete film-core arrays in a functional mode, and 3) to design and construct a feasibility model of a 1024-word memory. The desired memory had to be easily alterable by electrical means at a speed fast enough to permit its being loaded from paper or magnetic tape. It also had to have random-access readout, high reliability, simplicity of design and still retain the characteristic advantages of other thin-film memories.

Operating Principles

The BICORE memory elements used are

thin magnetic discs (possessing uniaxial anisotropy) that can be magnetized in either of two remanent states, and therefore are capable of storing information as binary digits. Two thin films are used for each memory element: one of a high-coercivity cobalt-iron alloy, and one of a low-coercivity nickel-iron alloy. The former is used to store the bit whereas the latter senses it. These film elements were vacuum deposited in a multilayer fashion onto a single glass substrate. The film cores are circular discs with a diameter of 0.050 in. They are placed on 0.100-in. centers and fabricated in 16 x 18 arrays.

Since the two types of films are in close proximity, the result is a film-pair from which data can be read out nondestructively. The Co-Fe film is called the "storage" film because it is the one that is switched, by a relatively strong applied field, to the desired state during the 'write' operation. The Ni-Fe film is termed "readout" film because it alone is switched during the 'read' operation and because its remanent state is determined by the state of the storage film. These properties are depicted in Fig. 1. The average external (demagnetizing) field of the storage film is sufficient to saturate the readout film in a direction such that the flux path between the two films is closed. When a "one" is stored, the read field is sufficiently large to overcome the external field of the storage film and switch the readout film to the opposite ("zero") state. This switching, indicating a stored "one", produces a voltage on a nearby sense line. The 'read' field is not large enough to produce any change in the storage film. As the 'read' field subsides, the external field of the storage film restores the readout film to its original ("one") state. When a "zero" is stored, the 'read' field aids the external field of the "storage" film and drives the "readout" film further into saturation ("zero" state); no output voltage is produced on the sense line.

Hysteresis loops of a BICORE memory element under three conditions are shown in Fig. 2. Figure 2(a) is for a high drive condition and the dual nature of the memory element can be readily seen. Figure 2(b) is for a low drive condition with the high-H_c, or storage film, film demagnetized. One can see that the resulting loop is centered, and is the result of the low-H_c film switching alone, since the drive field is not large enough to move any domain walls in the high-H_c film. Figure 2(c) is a loop of the low-H_c film, but with the storage film switched to one of its remanent states. It is interesting to note that the loop is no longer centered, but shifted by an amount equal to the external demagnetizing field of the storage film.

The basis of the memory operation is depicted in Fig. 3, which shows the stylized hysteresis loops for the films. The loops are not drawn to scale. The external field of the storage film is greater by one-third than the sum of the coercivity and average demagnetizing field of the readout film. The hysteresis loop of the readout film indicated in Fig. 3 is as it would be if the storage film were demagnetized. When the storage film is switched to one of its remanent states, however, the loop of the readout film is shifted or biased with respect to the drive field (to the right for a stored "one" and to the left for a stored "zero").

The minimum 'read' field that will switch the entire readout film, H_R (min), is equal to the sum of the external (demagnetizing) field of the storage film, the coercivity of the storage film and the external field of the readout film.

$$H_R \text{ (min)} = H_{Ds} + H_{Cr} + H_{Dr}$$

The maximum read field that will begin to switch the storage film, H_R (max), is equal to the sum of the coercivity of the storage film and the external field of the readout film less the external field of the storage film.

$$H_R \text{ (max)} = H_{Cs} - H_{Ds} + H_{Dr}$$

It can thus be seen that the read field need not be critically controlled.

One inherent advantage of the BICORE memory element, when operated in the described mode, is that the sense-line output voltage has a rather nonlinear response to the 'read' field. Consequently, small "sneak" currents from nonselected or partially selected word gates will not produce output signals that when added will be comparable with a "one" output. In contrast to this, many other techniques of achieving random-access NDRO produce output voltages that have either a linear or quadratic response to drive currents. Consequently, in these other types of memories of larger size, fairly complex word drivers would be needed to suppress "sneak" currents.

Figure 4 illustrates schematically the arrangement of a BICORE memory element and associated drive and sense conductors. As shown in the cross-section, the conductors form a loop around the film-planes in a "sandwich-type" construction. The etched-circuit array used for the conductors is more fully described in a later paragraph.

Two sets of conductors, the word line and the digit line, link each memory element and run in general directions at right angles to each other. At the memory element position, however, the conductors are parallel as shown in Fig. 4. The easy directions of magnetization of the film cores are nominally at right angles to the conductor segments so that fields produced by currents in the conductors are generally longitudinal.

The word line carries the 'interrogate' current during the 'read' operation and an 'address-select' pulse during the 'write' operation. The digit line is utilized during the 'read' operation for sensing and carries the digit-drive pulse during the 'write' operation. The nominal pulse schedule for writing and reading is shown in Fig. 5.

In the 'write' operation a bipolar pulse is passed down the word line, and a smaller pulse, which "brackets" the word pulse in time, is placed on the digit line. The polarity of this latter pulse determines what will be stored in that particular bit position of the selected word. This is the typical word-organized writing operation wherein the field used to switch the storage films is three times as great as the field from the digit-line current which alone does not switch the storage films.

During the 'read' operation, a pulse applied to the word line produces an output from those memory elements that contain "ones". The negative-current portion of the read current restores the word transformer and reduces the time needed to reset the readout films. This negative pulse can, under certain conditions, cause slight outputs from stored "zeros". In normal operation, however, the sense-amplifier output will not be enabled during the time the read current is negative. There is inductive coupling between the read and sense lines since they are parallel at the bit location. This coupling, which would produce a signal about equal to film output, must therefore be canceled by an opposite coupling.

Because of the manner in which current flows through the drive lines, the direction of fields near the elements are not quite perpendicular to the conductor segments; instead, the effective field is offset slightly. To compensate for this, and also to reduce switching time during the read operation, the easy directions of the films were rotated from the nominal direction shown in Fig. 4. The direction of this rotation was such that the field produced by the word-line current became more skewed from the easy direction while the field from the digit-line current was along the easy direction. Because of these effects, a slight rotation of the magnetization vector of the storage film from the "zero" state causes a small output signal, but calculations indicate it is small. Calculated values of optimum currents for reading and writing are also altered somewhat by this effect.

Model Memory

Design

A 288-bit model memory was constructed to verify the design using BICORE memory element technique, and to provide a device to rapidly test film-core arrays. The memory block diagram is contained in Fig. 6. The memory is word organized and consists of sixteen 18-bit words. A console switch is used to select either "write" or "read". During the 'write' mode, the content of the load register is stored in memory (by controlling the bit drivers) at the location contained in the address register. After all 16 words are loaded, the memory can be switched to 'read' and after each 'read' operation, the content of the 18-bit readout register is checked for errors. The error test (selected by the operator) is either a bit-by-bit comparison with the load register or an odd-parity check. An error causes an indicator to light and, if a two-position console switch is in the proper position, causes the memory to stop operating. The contents of the address register and readout register indicate which memory element contains the error. Auxiliary controls are used to determine limits of the writing and reading currents.

Circuits for the memory are contained on about 250 cards 2-1/2 in. by 2 in.

A disassembled plane of the model memory is shown in Fig. 7. The film array is sandwiched between the two etched circuits which are bolted together and plugged into the memory connectors.

Results

The model memory has been operated very satisfactorily. Arrays have been tested. It was found that write currents could be varied ± 10 per cent from nominal; and read currents can be typically varied over a 2 to 1 range without producing errors. Figure 8 shows typical waveforms of eight "ones" and eight "zeros" measured at the output of a two-stage linear amplifier. Switching times of the film outputs shown in this photograph are limited by the rise times of the word generator and the sense amplifier.

Memory planes were heated during operation. The only effect this high temperature had on the memory was a 5 per cent reduction in the upper limit of the read current; however, the useful range of read current is sufficiently large so that temperature compensation is not required. Several arrays have been read out up to 10^9 times for each memory element with no degradation of memory contents. Patterns were written into several arrays and these later were subjected to various environmental tests. These tests included high- and low-temperature storage and cycling, humidity, vibration and shock. In all cases, subsequent checks indicated no changes had occurred in stored patterns. Results of these tests indicate that the BICORE memory is a practical thin-film device capable of reliable, fast, nondestructive readout.

1024-Word Memory

General Considerations

Upon successful completion of the model memory described above and subsequent tests, the designing of a 1024-word 36-bit memory employing the same mode of operation was undertaken. Read cycle time was to be 1.5 μsec, write cycle time 100 μsec (mostly determined by duty-cycle considerations of semiconductors).

Because of the increased size over the smaller memory it was obvious that certain new approaches had to be taken. For example, in a linear selection memory of this type, it would not be practical to have 1024-word drivers packaged in the conventional manner with 1024 leads connecting to the stack. This is particularly true in light of the low impedance of the drive lines.

It was also felt that it would be very desirable to use a single word-gate for reading and writing: one that would match the low impedance of the word lines to that of semiconductor circuits.

Since this memory would employ 128 film-core arrays of 18 x 16 bits it was not practical to employ the type of plane assemblies used in the model memory.

Plane Design

To satisfy the above requirements, the memory plane shown in Fig. 9 was designed. It accommodates eight 18 x 16 arrays which are arranged to form sixty-four 36-bit words. Sixteen of these planes can be stacked to yield a 1024-word memory.

The plane uses the same geometry for the two layer etched circuit as the model memory. A "fold-over" is used to provide the end connections to the word line. The diode-core word gates are mounted on a component board and become an integral part of the plane assemble.

Word-Selection System

The word-selection scheme is shown in Fig. 10. The system utilizes 1024 permalloy wound switch cores and high-conductance diodes. Sixty-four switch cores and diodes are mounted on a component board on each of the 16 memory planes. The switch cores and diodes are arranged electrically in a 32 x 32 matrix. To select a word, one of the 32 X-selection switches is turned on and a short time later one of the 32 Y-select lines is pulsed. Current in the primary of the selected switch core overcomes a bias current and provides a positive current pulse down the word line. When the primary current is removed, the bias re-switches the core and provides a negative current pulse down the word line.

The dual-polarity word pulse is used in conjunction with an overlapping positive or negative digit pulse to accomplish writing a "one" or a "zero". The selection system functions in the same manner for both read and write mode. The proper amplitude of positive and negative current for read or write mode is obtained by adjusting the regulator and changing the bias current through the switch cores.

Stack Construction

The 1024-word store, shown in Fig. 11 was constructed by stacking 16 planes onto a base plate containing a set of vertical bolts to maintain proper alignment. A top plate that, when tightened down, applies pressure to clip connectors between each plane was attached. As a result of this arrangement, the digit lines are connected in series throughout the stack and the diode tabs are shorted in an appropriate manner to yield the arrangement of Fig. 10. The planes are separated by 1/6 in. and the over-all stack measures 6 x 9 x 12 in. Since the address-selection circuits are in the stack, a total of 32 + 32 + 72 leads leave the stack.

Results

The results of this memory were encouraging. Figure 12 shows voltage waveforms, measured at the test point of the sense amplifier, with 512 "ones" and 512 "zeros" stored. Although the outputs vary somewhat, the signal-to-noise ratio is quite adequate. Most of the variation in output is attributed to nonuniform film thicknesses and to imperfect registration within each memory plane. Currents could be varied by about ±15 per cent without producing errors.

Conclusions

BICORE memory elements offer a practical approach to a memory in which random access, high-speed NDRO operation is desired but write-in time is not critical. Results indicate that a memory of this type is insensitive to environmental conditions.

The fabrication techniques employed have considerable potential for size reduction and high-bit density.

Acknowledgements

The authors appreciate greatly the work of the people who assisted in this effort and the encouragement of those who directed it.

---

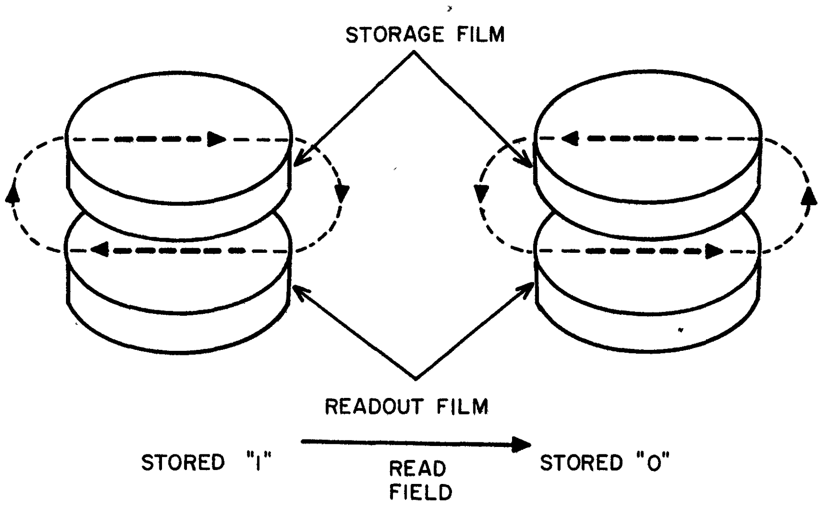
1 A. V. Pohm and S. M. Rubens, "A Compact Coincident-Current Memory", Proc. E.J.C. Conf., pp 120-123; Dec. 1956.

2 C. D. Olson and A. V. Pohm, "Flux Reversal in Thin Films of 82 Per Cent Ni, 18 Per Cent Fe", J. Appl. Phys., Vol.29, pp 274-282; March 1958.

3 D. O. Smith, "Magnetization Reversal and Thin Films", J. Appl. Phys., Vol. 29, pp 264-273; March 1958.

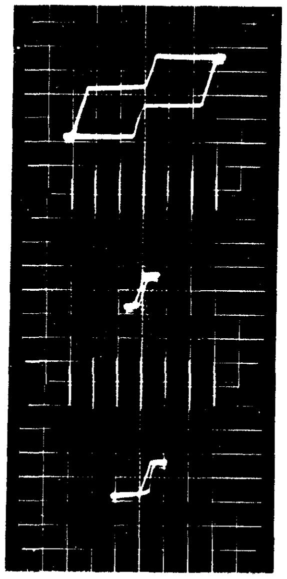
4 J. I. Raffel, "Operating Characteristics of a Thin-Film Memory", J. Appl. Phys., Vol. 30, pp 60S-61S; April 1959.

5 L. J. Oakland and T. D. Rossing, "Coincident-Current Nondestructive Readout from Thin Magnetic Films", J. Appl. Phys., Vol. 30, pp 54S-55S; April 1959.



Note:
Dotted arrows indicate direction in which films are magnetized. Direction of interrogating read field is shown by solid arrow.

Figure 1. COUPLING OF STORAGE AND READOUT FILM



a. High Drive

b. Low Drive with Storage Film demagnetized

c. Low Drive with Storage Film switched to a remnant state

Figure 2. BICORE MEMORY ELEMENT HYSTERSIS LOOP

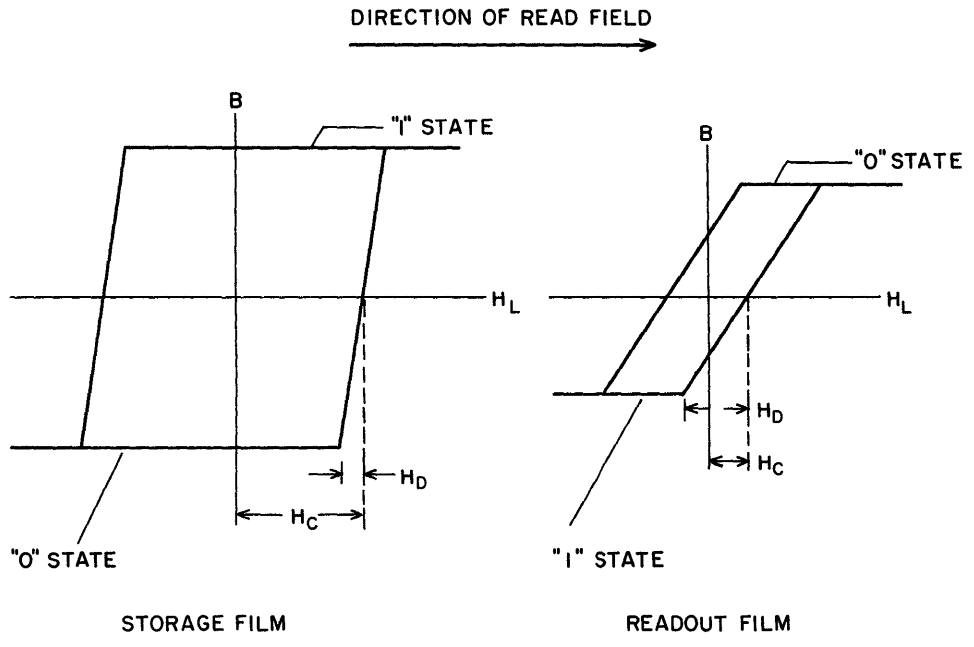
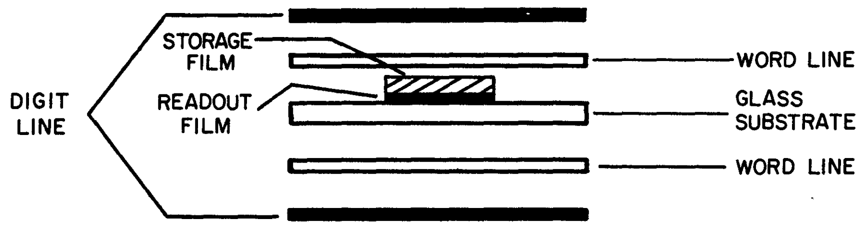
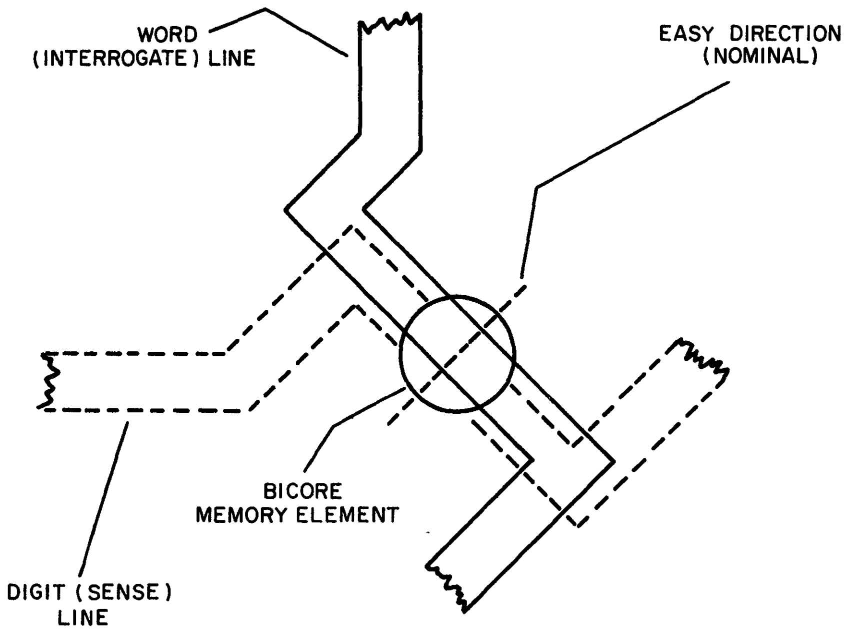


Figure 3. STYLIZED HYSTERSIS LOOPS



(a) CROSS SECTION SHOWING RELATIONSHIP OF FILM CORES AND ASSOCIATED CONDUCTORS



(b) GEOMETRY OF WORD DIGIT LINES (TOP VIEW)

Note:

The two lines, word and digit, run parallel in vicinity of memory element; both serve a double function. Each carries a current during write mode. During read mode, word line carries pulse for interrogating "readout" film and digit line carries output pulse.

Figure 4. SCHEMATIC ARRANGEMENT OF BICORE MEMORY ELEMENT

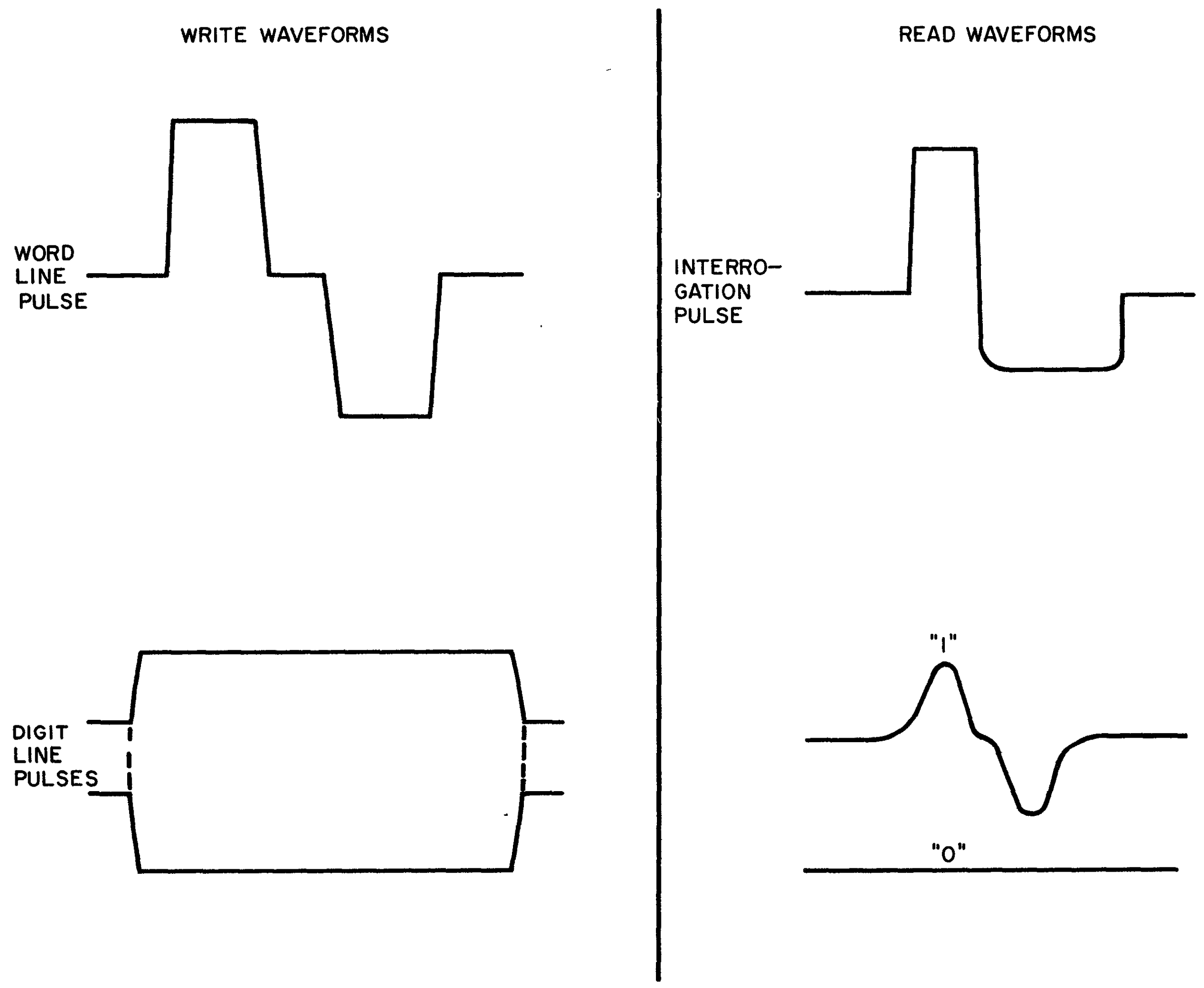


Figure 5. PULSE SCHEDULE

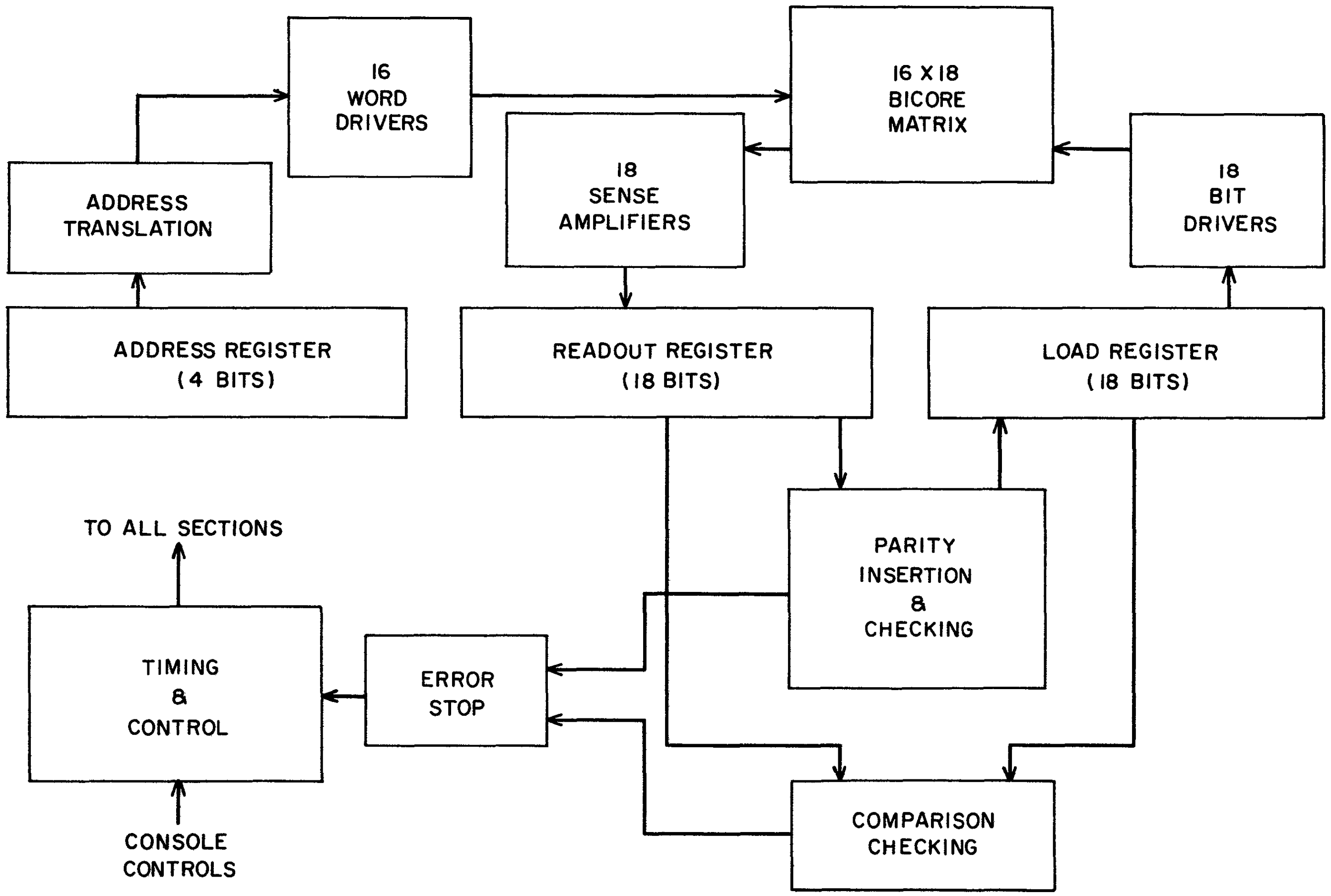


Figure 6. MODEL MEMORY, BLOCK DIAGRAM

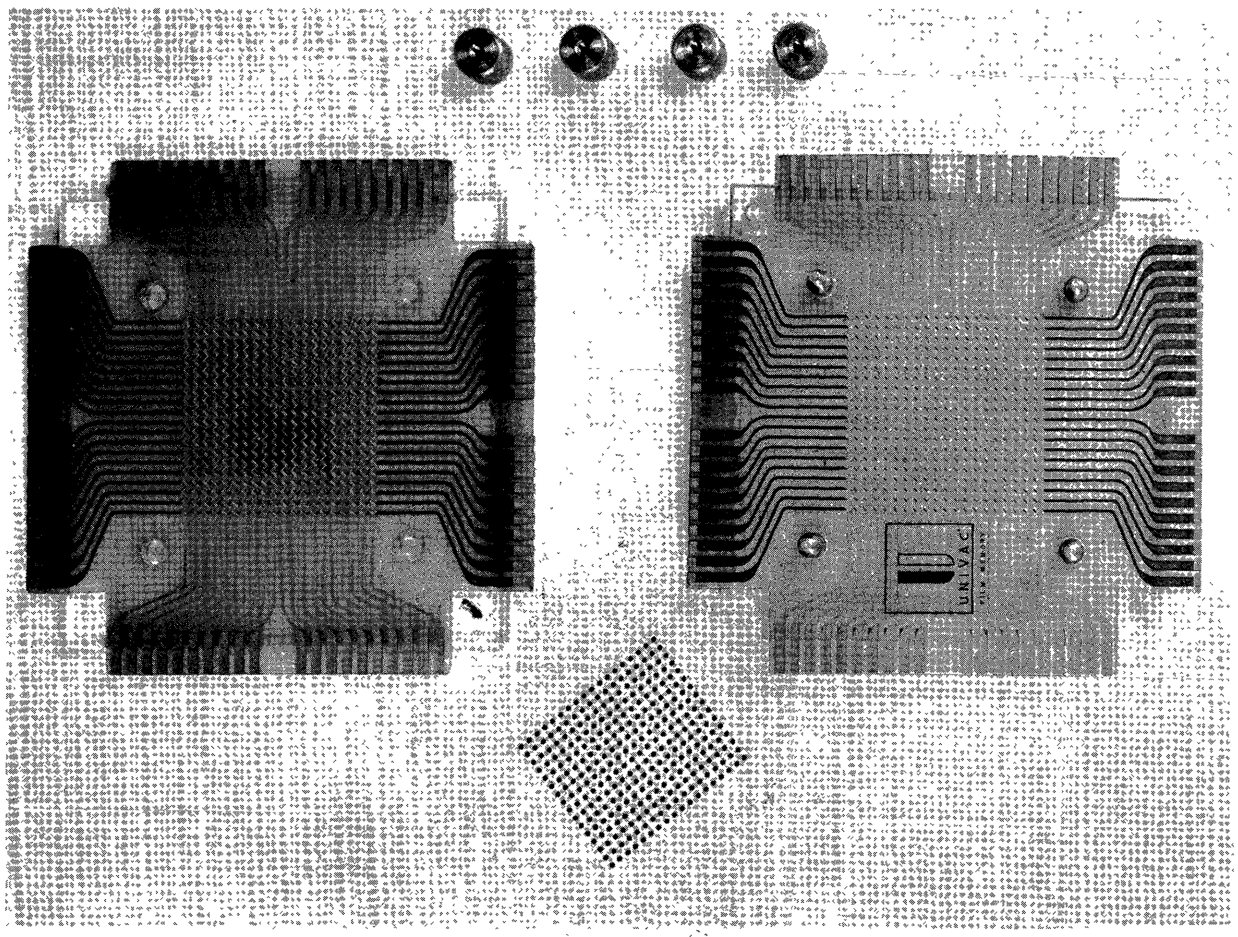
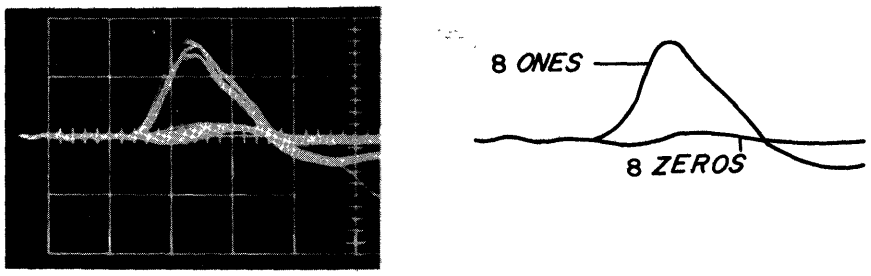


Figure 7. PLANE ASSEMBLY FOR MODEL MEMORY



VERTICAL CALIBRATION: 0.5 VOLT/DIVISION
HORIZONTAL CALIBRATION: 0.2 μ SEC/DIVISION

Figure 8. "ONES" AND "ZEROS" AS READ FROM MODEL MEMORY

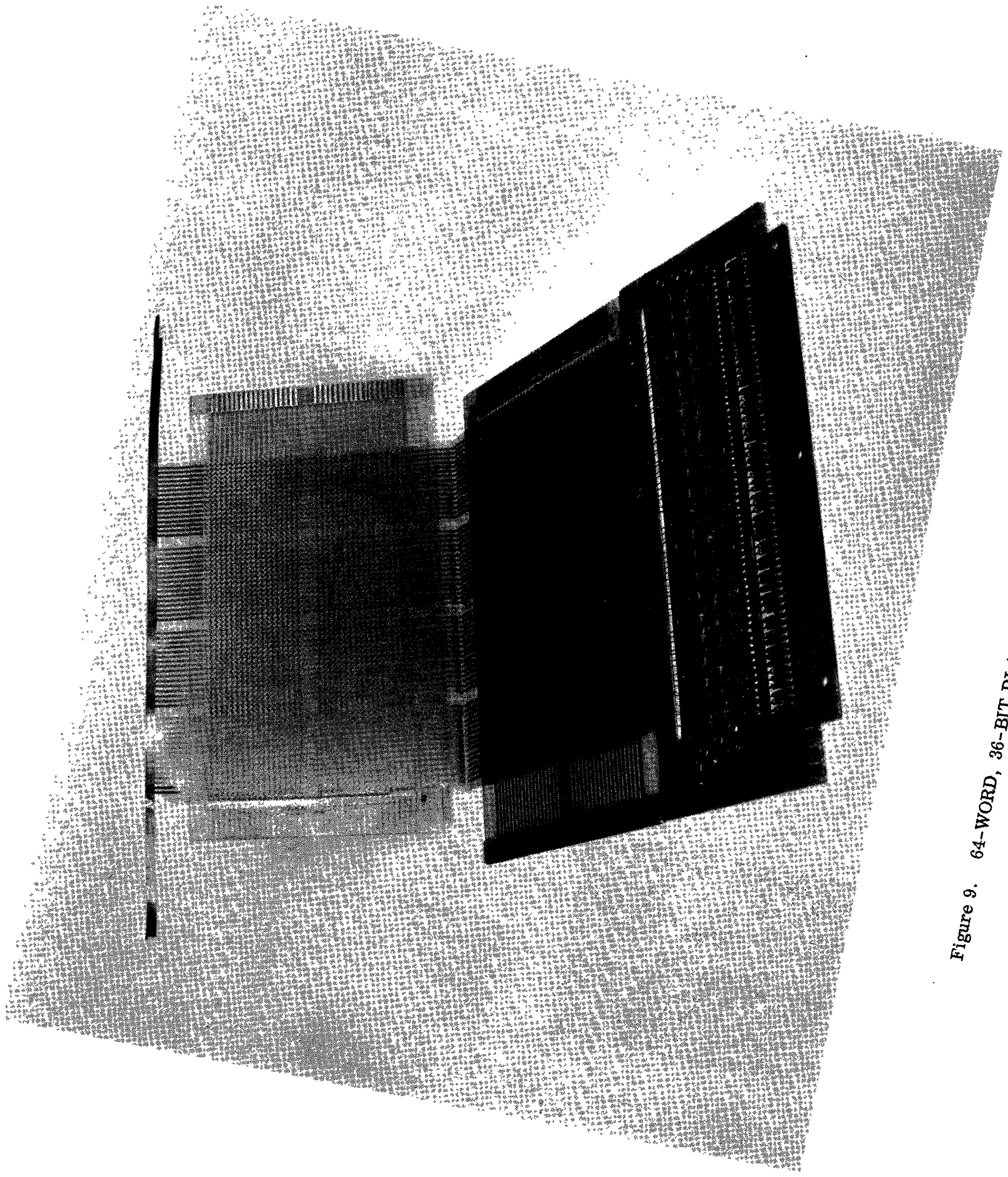


Figure 9. 64-WORD, 36-BIT PLANE ASSEMBLY

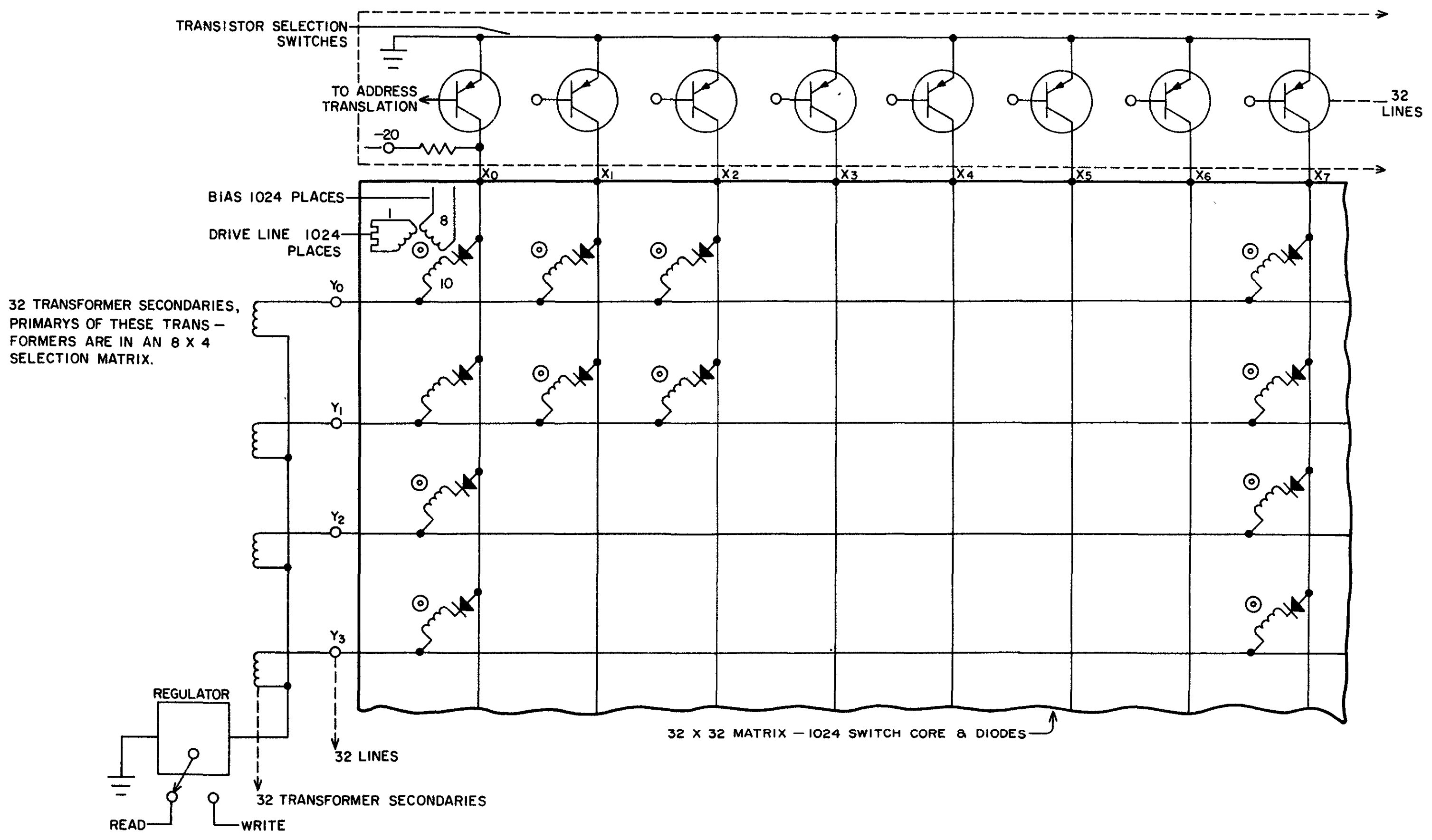

TRANSISTOR SELECTION SWITCHES
TO ADDRESS TRANSLATION
-20
32 LINES
X_0
X_1
X_2
X_3
X_4
X_5
X_6
X_7
BIAS 1024 PLACES
DRIVE LINE 1024 PLACES
1
8
10
32 TRANSFORMER SECONDARIES, PRIMARYS OF THESE TRANS-FORMERS ARE IN AN 8 X 4 SELECTION MATRIX.
Y_0
Y_1
Y_2
Y_3
REGULATOR
READ
WRITE
32 LINES
32 TRANSFORMER SECONDARIES
32 X 32 MATRIX — 1024 SWITCH CORE & DIODES

Figure 10. ADDRESS SELECTION SCHEME

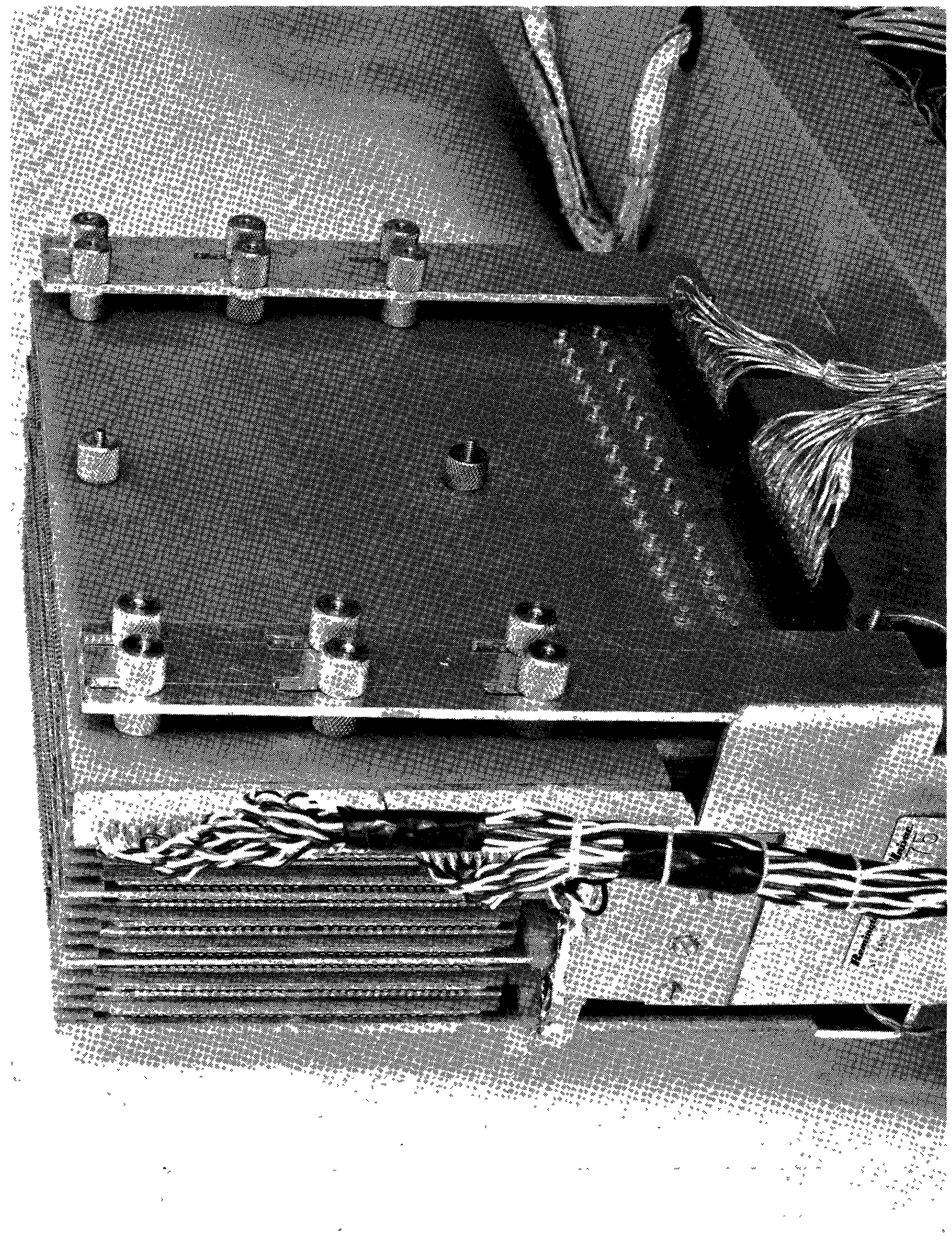


Figure 11. 1024 MEMORY STACK

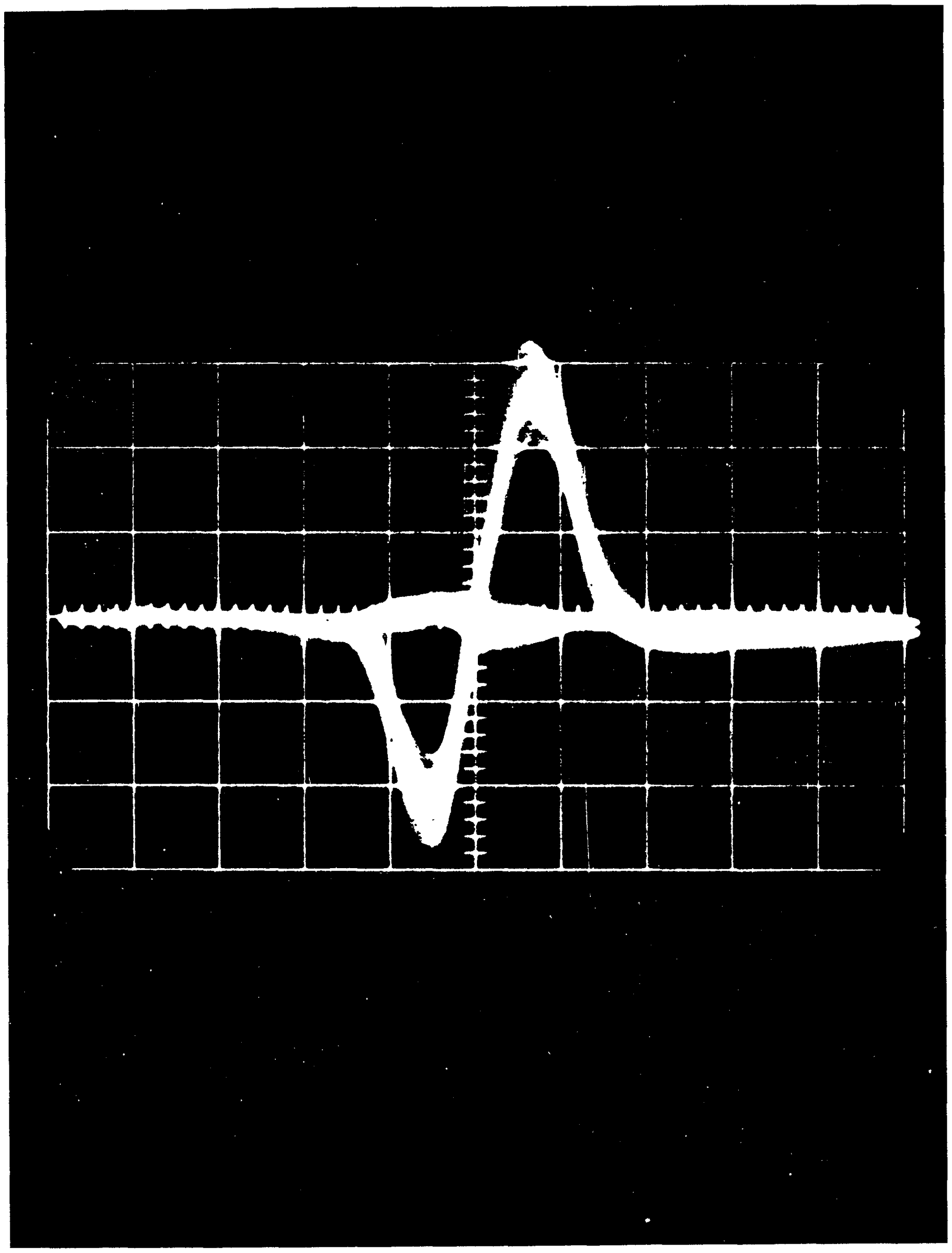


Figure 12. VOLTAGE OUTPUTS OF SENSE AMPLIFIER WITH 512 "ONES" AND 512 "ZEROS" STORED

TUNNEL DIODE STORAGE USING CURRENT SENSING

E. R. Beck, D. A. Savitt, and A. E. Whiteside

The Bendix Corporation
Research Laboratories Division
Southfield, Michigan

Summary

Tunnel diodes are attractive for use as the basic elements in high-speed random-access memories because of their fast switching speed and good environmental tolerance. A memory approach using tunnel diodes has been devised which is based upon destructive sensing of the operating current level in simple bistable elements. Each element consists of one tunnel diode and one resistor. This approach results in a near minimum of memory element complexity and of drive and bias power requirements. The immediate goal of the work reported is the development of a memory of somewhat over 1000 bits, able to operate from -55C to +125C at submicrosecond cycle times.

The feasibility of the overall memory concept, which is applicable to either bit or word organization, has been demonstrated by the operation of a test model. Test results indicate that a 200-ns cycle time is obtainable in a memory of 64 words of 24 bits each. Cycle time is roughly proportional to the square of the word capacity; correspondingly shorter cycle times can be obtained with memories of smaller capacity. With further circuit refinements it should be possible to operate a 64-word memory of this basic type at a cycle time of 100 ns or less over the temperature range of -55C to +125C.

Introduction

One of the most attractive immediate applications of tunnel diodes is in the construction of high-speed random-access memories.[1,2] The tunnel diode offers the possibility of operating such memories over wide temperature ranges. This paper discusses a memory element that has been devised to make efficient use of the tunnel diode. Since this element provides no isolation between drive currents and sense output, the form of the matrix has been arranged to provide the required isolation. The matrix form is explained, and the techniques employed for driving and sensing with this form of matrix are discussed. The paper ends with some test results and conclusions.

Basic Design Choices

Basic Memory Element Component

The tunnel diode is attractive for use as a memory element because of its fast switching speed and good environmental tolerance.[3] Potentially a low-cost device, the tunnel diode can be made with close initial tolerances on its parameters and on their variation with temperature. Tight packaging can also be used because of the tunnel diode's small size and low power consumption.

Semiconductor Types

With the exception of the tunnel diodes, only silicon semiconductor devices are employed in the circuitry. Either germanium or gallium arsenide tunnel diodes may be used in the basic memory elements; the switching speeds of the two types are comparable. Information received from manufacturers as well as tests of available units indicate that the necessary tolerances on parameters and their variation with temperature can be obtained. Gallium arsenide tunnel diodes are, however, capable of operation over a wider temperature range; roughly the same percentage of parameter variations occur from -55C to +150C as occur with germanium tunnel diodes from -55C to +100C. For this reason gallium arsenide units were adopted as soon as they became available. This choice must be qualified by the fact that the long-term stability of currently available gallium arsenide tunnel diodes is unsatisfactory. If this problem cannot be solved, germanium tunnel diodes must be used and the temperature range reduced.

System Organization

Word rather than bit organization was selected as being more suitable for the memory sizes of immediate concern and for the wide operating temperature range desired. It is possible to make a bit-organized memory using the concepts and circuits to be discussed, and in fact the feasibility of doing this was verified by the operation of a test model.[4] However, no saving in selection and drive circuitry results from using bit rather than word organization in small memories, and the increased system tolerances afforded by the latter are clearly helpful in meeting the temperature requirements. Destructive readout was chosen because it was felt that the additional memory-element complexity required for a nondestructive readout capability was not justified by the somewhat higher speed obtainable.

The overall system organization used is the same as that of any word-organized memory with destructive readout. The system block diagram is shown in Figure 1 for reference purposes. It may be noted that for REWRITE operations the sense outputs are brought directly to the bit-line drivers to avoid the propagation delay of the input-output register.

Memory Element

Destructive sensing of operating current level was chosen for the readout mechanism. This type of sensing allows the use of a bistable element with only two terminals and only two components, a tunnel diode and a resistor. Besides having the least number of components, this element allows the simplest matrix topology.

The form of the memory element is shown in Figure 2. The values of the bias voltage V_{BB} and the resistor are chosen to make the circuit bistable. The state of the element is changed by increasing or decreasing, as appropriate, the voltage across the element to force operation to the desired stable state. The element voltage is changed by applying at points X and Y drive pulses of the type indicated.

With the diode polarity shown, a negative READ pulse applied at Y is used to switch the element to its high-current state, defined as the ZERO state. The increase in element current which occurs if a ONE is read is detected with the use of the transformer shown in Figure 2. A separate transformer is not required for each element. One transformer is actually shared by all the elements in a given bit position of all words.

Switching of the element to its low-current state for writing of a ONE is done on the coincidence of a positive pulse at Y and a negative pulse at X. To obtain larger tolerances with the WRITE operation the d-c bias is offset in the direction of the READ switching voltage. This bias offset also reduces the d-c power consumption, and permits the two opposite-polarity drive pulses at the Y terminal to be of nearly equal magnitude. The latter simplifies the generation of these drive pulses.

An inherent property of the element shown is the lack of drive-sense isolation. Currents produced by the drive pulses will produce noise signals in the secondary of the sense transformer which may be difficult to distinguish from the element output. Successful use of this type of element depends upon the cancellation of drive current obtained with the matrix configuration to be described below.

Memory Matrix

Matrix Form

The basic memory elements are connected in the matrix form shown in Figure 3 for word-organized selection of n words of k bits each. Each vertical matrix line is a word line and is connected to the Y terminals (see Figure 2) of all elements of a given word. Each horizontal matrix line is a bit line and is connected to the X terminals of all elements of a given bit position.

A single common sense transformer is shown in each bit line. The drive pulses of Figure 2 are supplied by the generators shown schematically in each line of the matrix. Only the generators in the selected lines are active, and the remainder may be considered to be shorted out.

Memory Element Equivalent Circuit

To understand how the driver currents are cancelled to obtain drive-sense isolation, it is helpful to consider an equivalent circuit for the memory element. Figure 4 shows that the memory element (a) may be represented by the equivalent circuit (b) consisting of a resistor r in shunt with a current generator I_s. Resistor r is the element incremental resistance, and I_s is the difference in element current in the two stable states.

The justification for the equivalent circuit is illustrated by Figure 4(c). The memory element I-V characteristic is shown along with corresponding time plots of current and voltage waveforms. Assume that the element operating point is initially at point A. As the voltage across the element is increased from V_{BB} to V_H, the current through the element suddenly drops as switching from the high-current state to the low-current state occurs. This current change may be represented by the closing of the switch in series with the current generator in the equivalent circuit. The current generator provides a step of current which, if the element is allowed to remain in its new state, results in a change in the current drawn from the d-c bias supply. Similarly, an equivalent circuit for the element can be derived for the case where the initial operating point is at B and the voltage is reduced below V_L; this circuit differs from (b) only in the polarity of the generator I_s. If the voltage across the element is between V_L and V_H the element is simply represented by resistor r.

Drive-Current Cancellation

The drive-current cancellation can now be explained by referring back to the matrix of Figure 3. Current from the selected word-line driver is cancelled out of the sense transformers by simultaneous operation of the cancellation driver feeding the dummy "cancellation word" (resistors r) on the opposite side of the transformer primaries. Cancellation driver and word-line driver waveforms are identical. Selection of the proper cancellation word is made from the most significant bit of the address specified.

Current from the bit-line drivers is cancelled out of each sense transformer by applying this drive to the center tap of the primary winding. These transformers are placed in the center of the bit lines, with half of the word lines on each side of the transformer primaries. By replacing the elements with their equivalent circuits, it can be seen that the primaries are symmetrically loaded and the bit-line drivers produce no net current in the sense transformers.

Matrix Equivalent Circuits

For purposes of explaining the sense and driver requirements, the matrix may be replaced with the simple equivalent circuits shown in Figure 5. The matrix presents a load to the word-line and cancellation drivers equivalent to a resistor of value r/k as shown in Figure 5(a). The matrix presents a load to the bit-line driver equivalent to a resistor of value r/n as shown in Figure 5(b). Therefore the loading on the word-line driver is proportional to the number of bits per word, and the loading on the bit-line driver is proportional to the number of words.

The equivalent circuit of the matrix as a signal source for each sense circuit is shown in Figure 5(c). The current generator I_s of the memory element being switched is shunted by the incremental resistance of all n memory elements on the bit line. The equivalent sense transformer shown has a turns ratio of 1:2a; the turns ratio of the center-tapped transformer actually used is 1:a. The ratio 1:a is chosen for maximum transfer of power to the sense circuit. It can be seen that the matrix output power is inversely proportional to n, the number of words. Note also that the polarity of the I_s generator will depend upon the location of the selected word; that is, to which end of the sense transformer primary the element is connected.

One of the main determinants of cycle time in a destructively-read memory is the delay in the rewrite loop. It can be derived from the matrix equivalent circuits, Figure 5(b) and (c), that the power gain required in the rewrite loop, and hence the rewrite loop delay, is proportional to n^2. The memory cycle time then varies roughly in proportion to the square of the memory word capacity.

Drive Circuitry

Coupling to Matrix

The drive generators shown in the matrix of Figure 3 were indicated for ease of explaation. The drive voltages are actually introduced in series with the bias and memory elements by means of transformers as shown in Figure 6. The drive voltages required are low, on the order of 0.9 volt, and the required tolerances cannot be obtained directly with silicon transistor and diode logic circuitry at such levels. The drive transformers provide an efficient voltage step-down from convenient, easy-to-control logic signal levels to the matrix drive levels. Simultaneously, this approach provides the necessary low driver a-c output impedances.

Drive Waveforms

Two cycles of the required drive-pulse waveforms are shown in Figure 7. These essentially repeat the waveforms shown in Figure 2 except that an additional bit-line drive pulse is indicated with dotted lines. Since word-organization is being employed, reading is accomplished by the negative word-line pulse alone. Writing a ONE, however, requires the coincidence of the word-line and bit-line pulses. Writing a ZERO is accomplished by delaying the bit-line pulse so that it does not coincide with the word-line pulse. The delayed bit-line pulse is shown with dotted lines in Figure 7.

A delayed bit-line pulse must be used when a ZERO is being written because the drive voltages are a-c coupled to the matrix lines. If a pulse were not applied on the bit line every cycle time, whether a ZERO or a ONE was being written, the pulse duty cycle would be variable and dependent upon the sequence of memory operations. Such a variable duty cycle would make the effective matrix bias voltage variable, reducing the bias-supply tolerances. Note that by employing equal-area read and write pulses on the word line the average value of this waveform can be made equal to zero for each cycle; therefore there is no duty cycle problem on this line.

Drive Circuits

The pairs of alternate-polarity drive pulses required for the word lines are obtained by differentiating a rectangular OPERATE pulse provided by the word-line pulse generator. (See Figure 1). Only unipolar pulses need then be handled by the word-selection circuitry. Each word-line driver consists of a tuned transformer driven by a diode AND gate as shown in Figure 8. The transformer is critically damped and the write pulse amplitude is clamped. The last level of address decoding is also done in this circuit.

The bit-line driver is shown in Figure 9. Transistor Q_1 is a pulse amplifier and transistor Q_2 provides low-frequency feedback for control of the pulse area. A clamp voltage is used to control the pulse amplitude, and resistor R_T provides temperature compensation. The input to this circuit is supplied from a three-input OR gate which in turn is fed from three AND gates. The AND gates are controlled by the sense circuit outputs, the input-output register and the OPERATE READ command. (See Figure 1).

Sense Circuitry

Level Restoration

The sense circuit must detect the presence of and amplify the current step I_s (of the matrix equivalent circuit) produced by a switching memory element. Each time another element switches, another current generator can be considered to be added in shunt with the one shown in the equivalent circuit of Figure 5(c). The level of the matrix output at any time then will vary, depending upon the polarity and spacing of the previous output step waveforms. It is therefore necessary to restore the matrix output level to zero after each new step of current occurs.

Level restoration is achieved in a manner similar to differentiation by the use of a short-circuited delay line connected across the sense transformer secondary. The manner in which level restoration is accomplished with a short-circuited delay line is illustrated in Figure 10. In this figure, e_g and R_g represent the equivalent circuit of the matrix as seen from the sense transformer secondary, and R_L represents the input impedance of the sense circuitry. The value of the parallel combination of R_g and R_L is made equal to the delay-line characteristic impedance R_o. Step waveforms from the matrix are then converted to rectangular pulses as shown. The pulse width is equal to twice the delay time T_d of the line. In general, a rectangular pulse is generated for each step of matrix output, and the pulse polarity is the same as the step polarity. The effect of the circuit, then, is to restore the matrix output level to zero before the next signal arrives. This permits an amplitude level discriminator to determine if a signal is present.

Level Discrimination

In addition to the step current waveforms resulting from the switching of memory elements, the matrix also produces some noise. Noise results from the imperfect cancellation of the drive currents in the matrix. The noise resulting from the word-line driver is relatively low compared to the matrix output signal level, because the cancellation element and driver can be matched closely to the active words and drivers.

The noise produced by the bit-line drivers, however, depends upon the match between the incremental resistances of all elements on one side of the sense transformer primary and the resistances of all elements on the other side. The noise level then is proportional to the matrix size in words and can be significant compared to the matrix signal for memory sizes of interest. However, this noise is produced only during the WRITE operation, when sensing of the matrix output is not performed.

Because of the presence of noise in the matrix output, it is desirable to use a level discriminator in the sense circuits. Since it is undesirable that a response be generated during the WRITE operation (because this could result in a recovery time problem at short cycle times), the discriminator is strobed during the READ operation. Since the pulses produced by the level restorer may be of either polarity, a bipolar level discriminator is required. The circuit employed to meet these requirements is shown in Figure 11(a).

The bipolar discriminator circuit is essentially two tunnel diodes in parallel, biased by a current source I_B to a point just below their peak currents. The output voltage is initially low. An input pulse is fed to the diodes through the center-tapped transformer. As a result a positive pulse appears in series with one tunnel diode and a negative pulse is in series with the other. If the input pulse exceeds the discrimination level, both diodes switch to their high-voltage state, and the output voltage goes high. Strobing is accomplished by using a pulse bias which is turned on only during the read operation.

The circuit operation is explained by Figure 11(b) and (c). Figure 11(b) shows the situation when the input voltage e_p is zero. The solid curve is the I-V characteristic of one diode, and the dotted curve is the characteristic of the two diodes in parallel. With the bias I_B shown, the initial operating point is A. Figure 11(c) illustrates the effect of e_p. The characteristic of one diode is shifted to the left and the other one to the right. The characteristic of the two in parallel, shown by the dotted line, then has a lower peak current than when e_p is zero. If e_p is large enough, the equivalent peak current is reduced below I_B and the operating point switches to B, producing an output signal.

Amplification

In addition to the functions of level restoration and discrimination, the sense circuits must provide amplification. Because of the shunting effect of all the other memory elements of a bit line on the output of the one memory element that switches, the amount of gain necessary to allow the sense output to operate the input-output register increases with the memory size.

To sense the low-power matrix output, a sensitive discriminator is required. Discriminator sensitivity is inversely proportional to the peak current of the discriminator diodes. Accordingly, low-peak-current units are employed. The discriminator is followed by two stages of tunnel-diode amplifiers of the analog threshold OR-gate type; successive stages use diodes having higher peak currents. All stages are biased by the same pulse supply, and interstage coupling is accomplished with high-speed diodes.

The switching time of a tunnel diode is approximately inversely proportional to the diode peak current. As a result, the sense delay contributed by the discriminator is proportional to the sensitivity. It was found for the 64-word memory that the delay with a single-stage video transistor pre-amplifier before the discriminator was less than that with a discriminator sensitive enough to operate directly on the matrix output. The discriminator employs two 0.5-ma tunnel diodes, the succeeding stage uses a 4.7-ma diode, and the output state employs a 22-ma diode.

Experimental Results

The feasibility of the concepts and circuitry described has been proved with a trial system which simulated a memory of 64 words of 24 bits each. Two words of active memory elements were used, and the remainder were simulated by resistors. The first tests were made with two active word-line drivers and one complete rewrite loop consisting of a bit-line driver and sense circuit. The loading and source-impedance effects of the remainder of the drivers were simulated.

The memory was operated in a repetitive READ-ONE, WRITE-ONE sequence at a cycle time of 280 ns. The rewrite loop was closed and a bit was circulated in a READ-ONE, REWRITE-ONE sequence at the same speed. Based upon the measured delays in the rewrite loop, it is estimated that the present circuitry is capable of running at a 200-ns cycle time. The additional delay in the 280-ns cycle was due to limitations in the word-line pulse generator.

Figure 12 shows driver and sense-output waveforms for a READ-ONE, REWRITE-ONE sequence; the waveforms are identical for a READ-ONE, WRITE-ONE sequence. (The presence of a sense output pulse indicates that a ONE has been read during this cycle by the word-line READ (negative) pulse. Since the bit-line driver pulse is in coincidence with the word-line WRITE (positive) pulse, a ONE is being written at this time.) The waveforms show a 60-ns delay in the sense circuits and a 40-ns delay in the rewrite logic.

Figure 13 shows the matrix and sense circuit outputs for both READ-ONE, WRITE-ONE (a) and READ-ZERO, WRITE-ZERO (b) repetitive memory operations. Two memory cycles are shown and the word-line driver waveform is included for a time reference. The matrix-output waveform includes the effect of the sense-circuit level restorer. A comparison of the matrix output waveforms of (a) and (b) shows the excellent ONE-to-ZERO ratio achieved. As stated before, the output level is inversely proportional to the number of words in the matrix. For the 64-word memory being operated here, a ONE produces a 20-mv pulse across the 250-ohm sense-circuit input impedance.

Limited temperature tests have been performed on the driver circuitry, and the results indicated that the tolerances required on the drive pulses can be met over the temperature range -55C to +125C. Figure 14 is a photograph of the test system chassis which consists of a base ground plane into which circuit cards are plugged. Two cards are shown in place, one of which is a digit plane card. A partially assembled digit plane is shown in the foreground.

Conclusions

The overall memory concept used, which is based on current sensing, appears to be one feasible approach to construction of a high-speed random-access memory. The approach results in low drive and bias power requirements. In the model discussed, the maximum d-c power consumption per bit is 1 mw; the maximum peak drive power per bit (for writing a ONE) is 1.6 mw. A cycle time of 200 ns is obtainable for a memory of 64 words of 24 bits each with present transistor driver circuitry. The cycle time varies roughly with the square of the number of words; correspondingly shorter cycle times can be obtained with smaller memories. The circuitry has been designed to operate from -55C to +125C, and uses only silicon diodes and transistors and gallium arsenide tunnel diodes.

The simple form of the memory element allows a high degree of matrix miniaturization. To show what can be done, the 16 by 16 array shown in Figure 15 was constructed, using tunnel diode packages expected to be available and small metal-film resistors.

Replacement of transistor drivers with tunnel-diode circuits is an attractive possibility. Drive pulses of the required amplitude can be obtained with a single tunnel diode switching between its low- and high-voltage states. Tunnel diode driver circuitry is simpler and more compact than the transistor-diode circuitry currently used. It should also be faster and, in particular, should greatly decrease the delay in the rewrite loop. It is believed that a 64-word memory will then be capable of cycle times of 100 ns or less from -55C to +125C.

Acknowledgement

The authors are indebted to fellow workers at Bendix Research Laboratories for contributions to the developments discussed in this paper. In particular, the authors would like to acknowledge the direction and encouragement provided by Dr. E. C. Johnson and Mr. R. C. Sims.

References

1. J. C. Miller, K. Li, and A. W. Lo, "The Tunnel Diode as a Storage Element," Digest of Technical Papers, International Solid-State Circuits Conference, Philadelphia, Pa., pp. 52-53, February 10-12, 1960.

2. M. M. Kaufman, "A Tunnel Diode Tenth Microsecond Memory," 1960 IRE International Convention Record, pt. 2, pp. 114-324.

3. R. N. Hall, "Tunnel Diodes," IRE Trans. On Electron Devices, Vol. ED-7, pp. 1-9, January, 1960.

4. R. C. Sims, E. R. Beck, Jr., and V. C. Kamm, "A Survey of Tunnel-Diode Digital Techniques," Proceedings of the IRE, Vol. 49, No. 1, pp. 136-141, January, 1961.

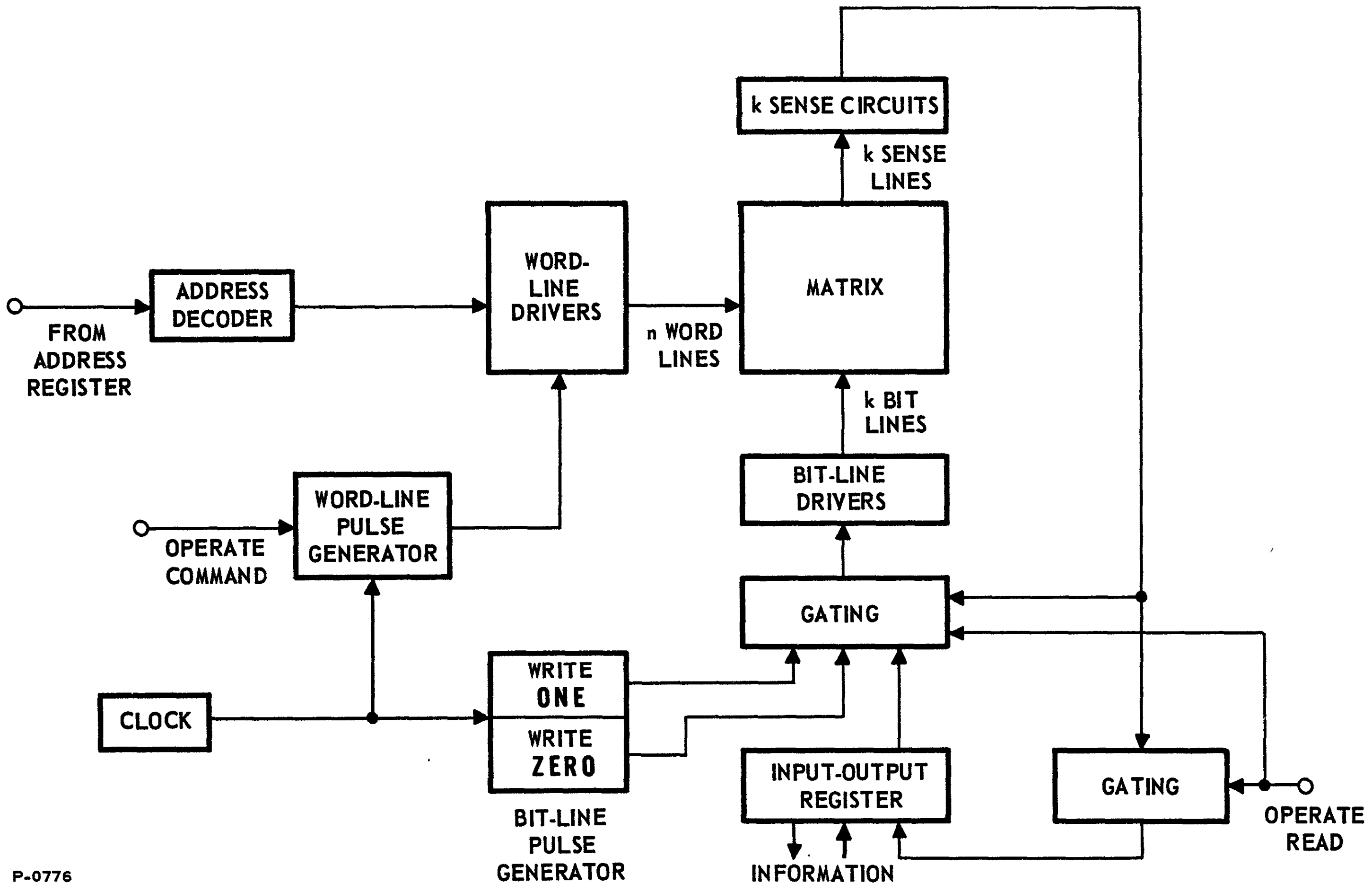


Figure 1. MEMORY SYSTEM BLOCK DIAGRAM

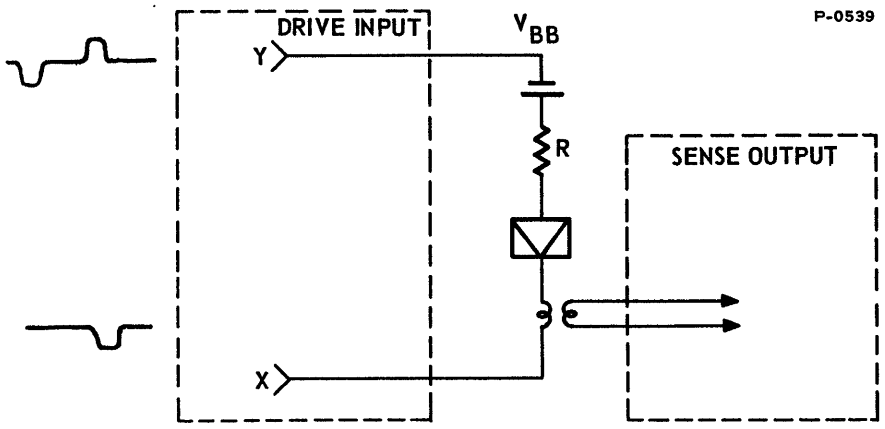


Figure 2. BASIC MEMORY ELEMENT

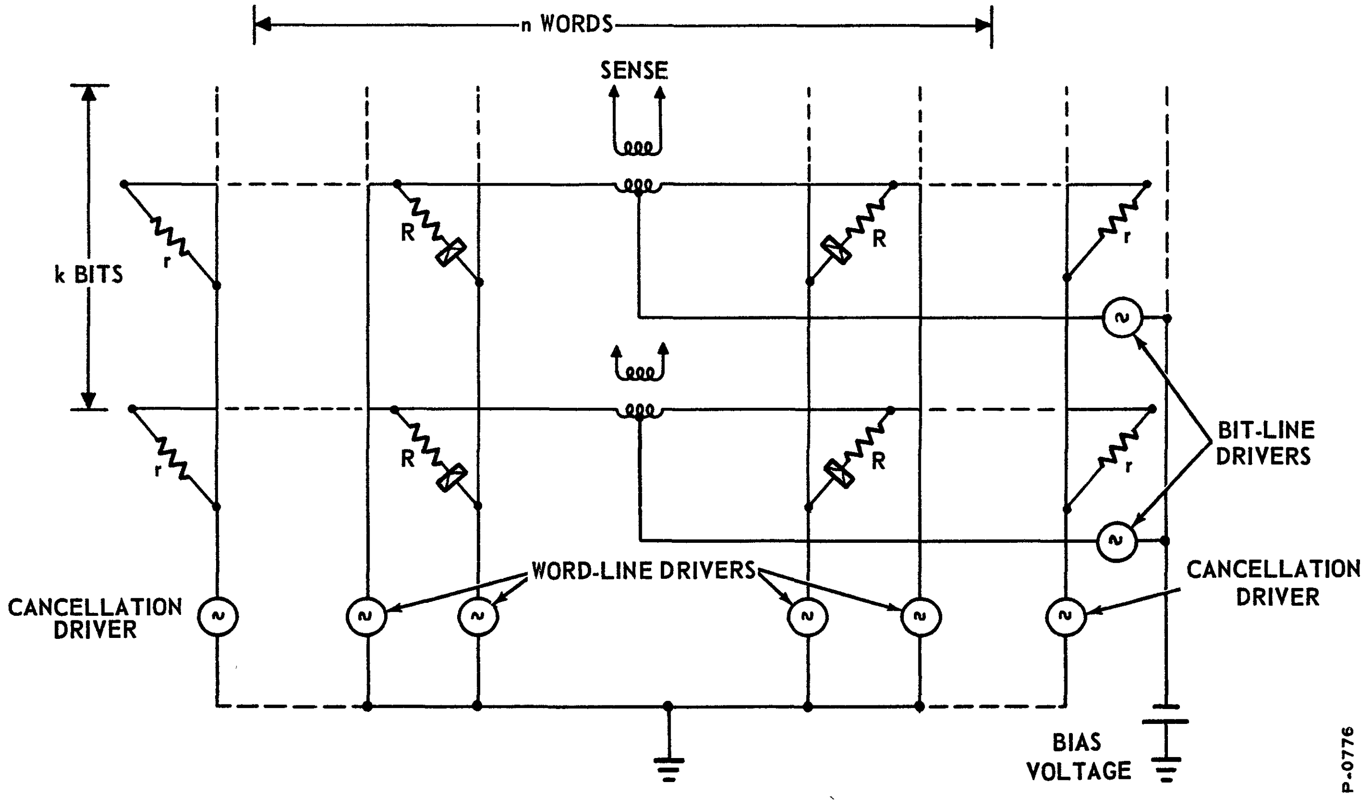


Figure 3. SCHEMATIC OF MEMORY MATRIX

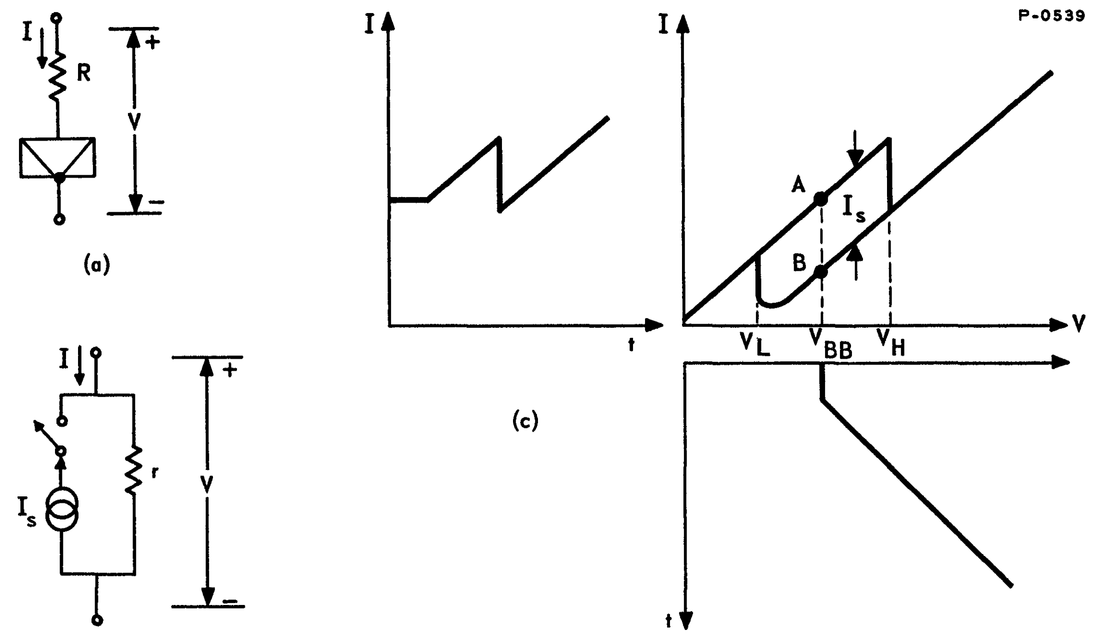


Figure 4. MEMORY ELEMENT EQUIVALENT CIRCUIT

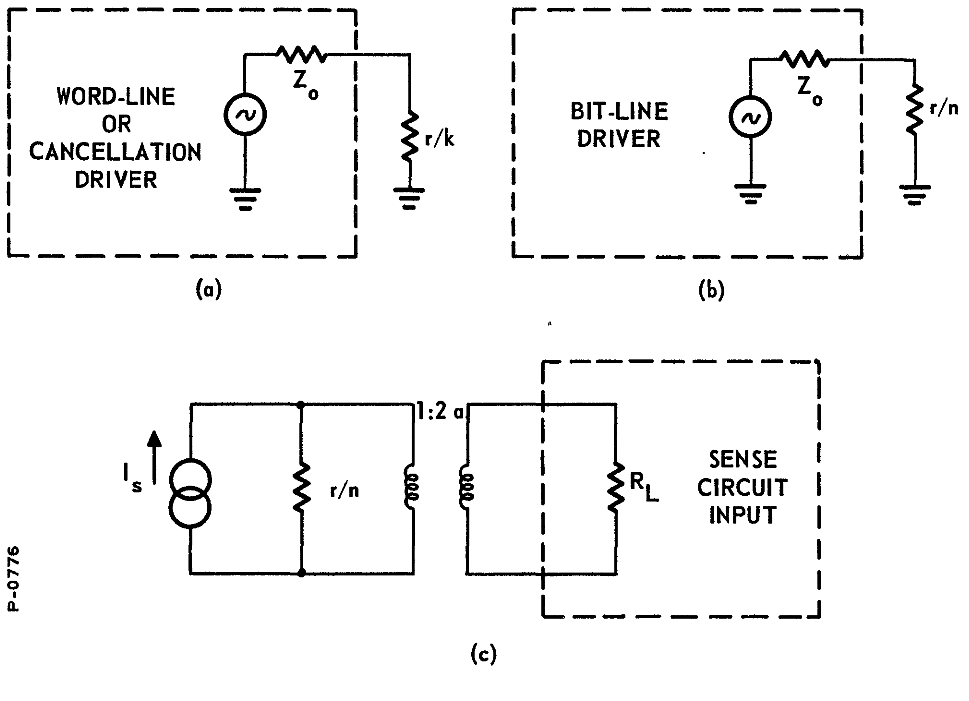


Figure 5. MATRIX EQUIVALENT CIRCUITS

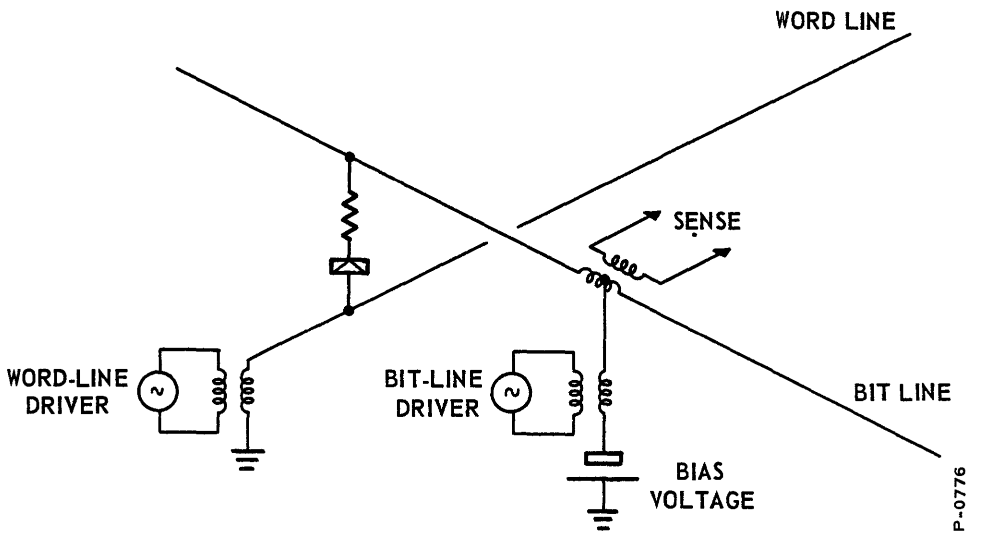


Figure 6. SCHEMATIC OF BIAS AND DRIVE ARRANGEMENT

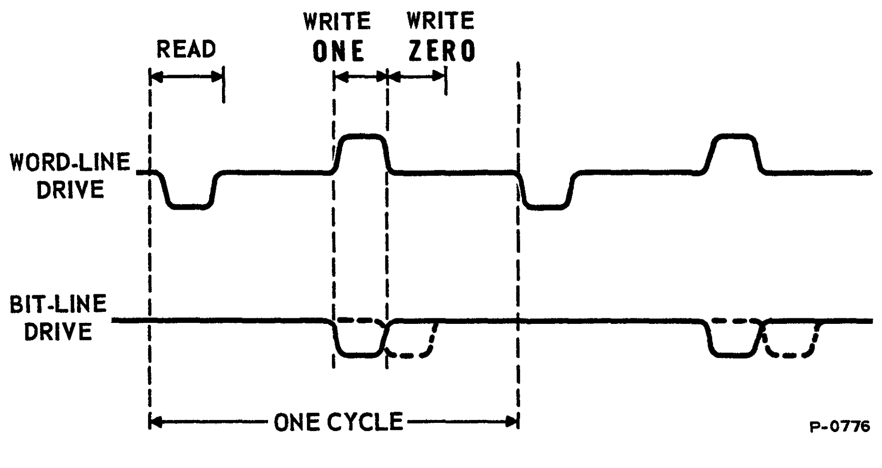


Figure 7. IDEALIZED DRIVE WAVEFORMS

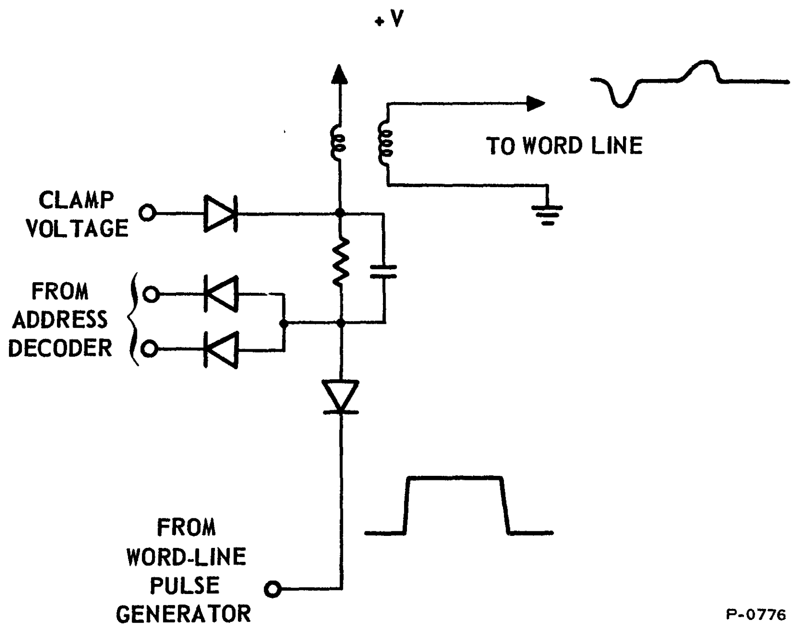


Figure 8. WORD-LINE DRIVER

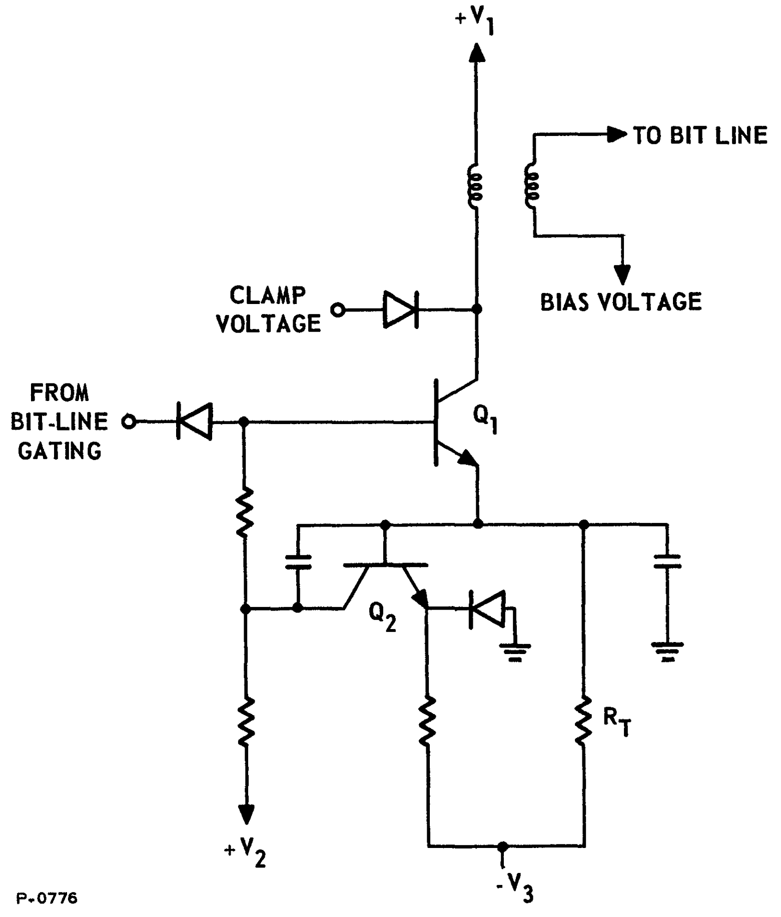


Figure 9. BIT-LINE DRIVER

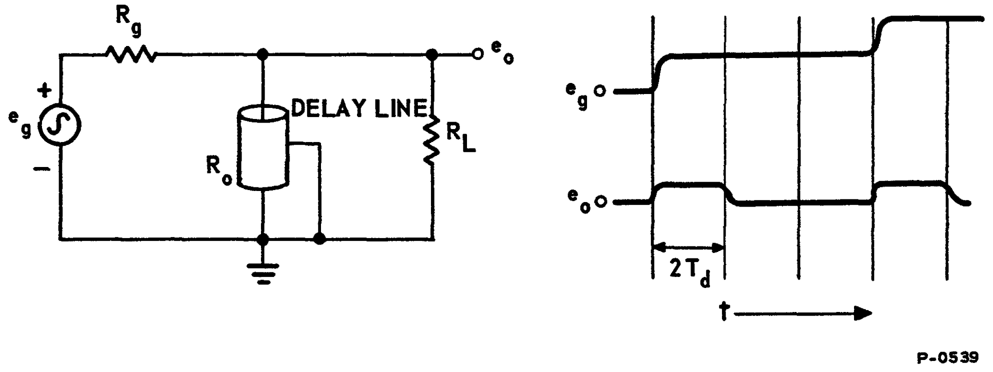
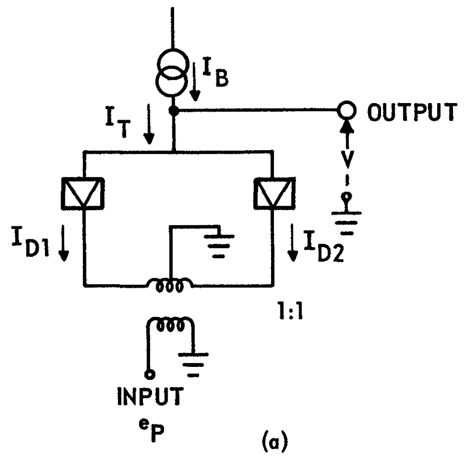


Figure 10. DELAY-LINE LEVEL RESTORER AND WAVEFORMS



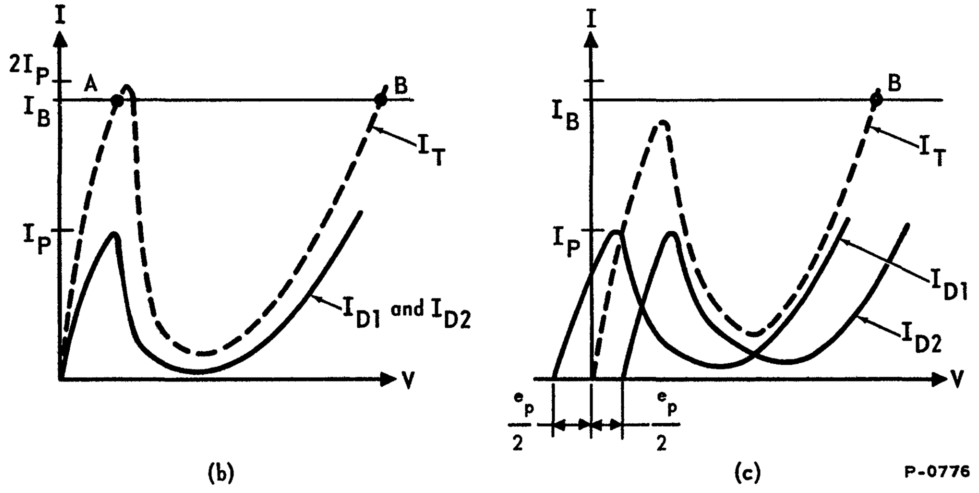


Figure 11. BIPOLAR DISCRIMINATOR

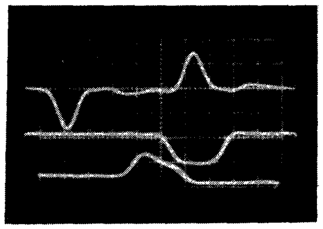


Figure 12. DRIVE AND SENSE-OUTPUT WAVEFORMS

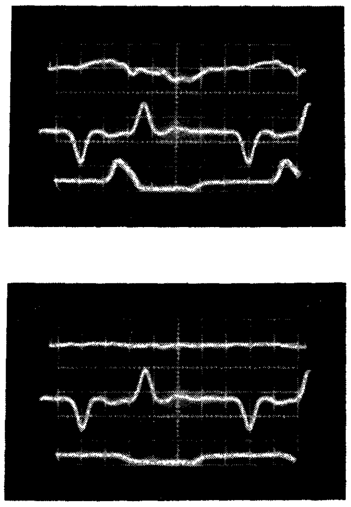


Figure 13. MATRIX AND SENSE CIRCUIT OUTPUT WAVEFORMS

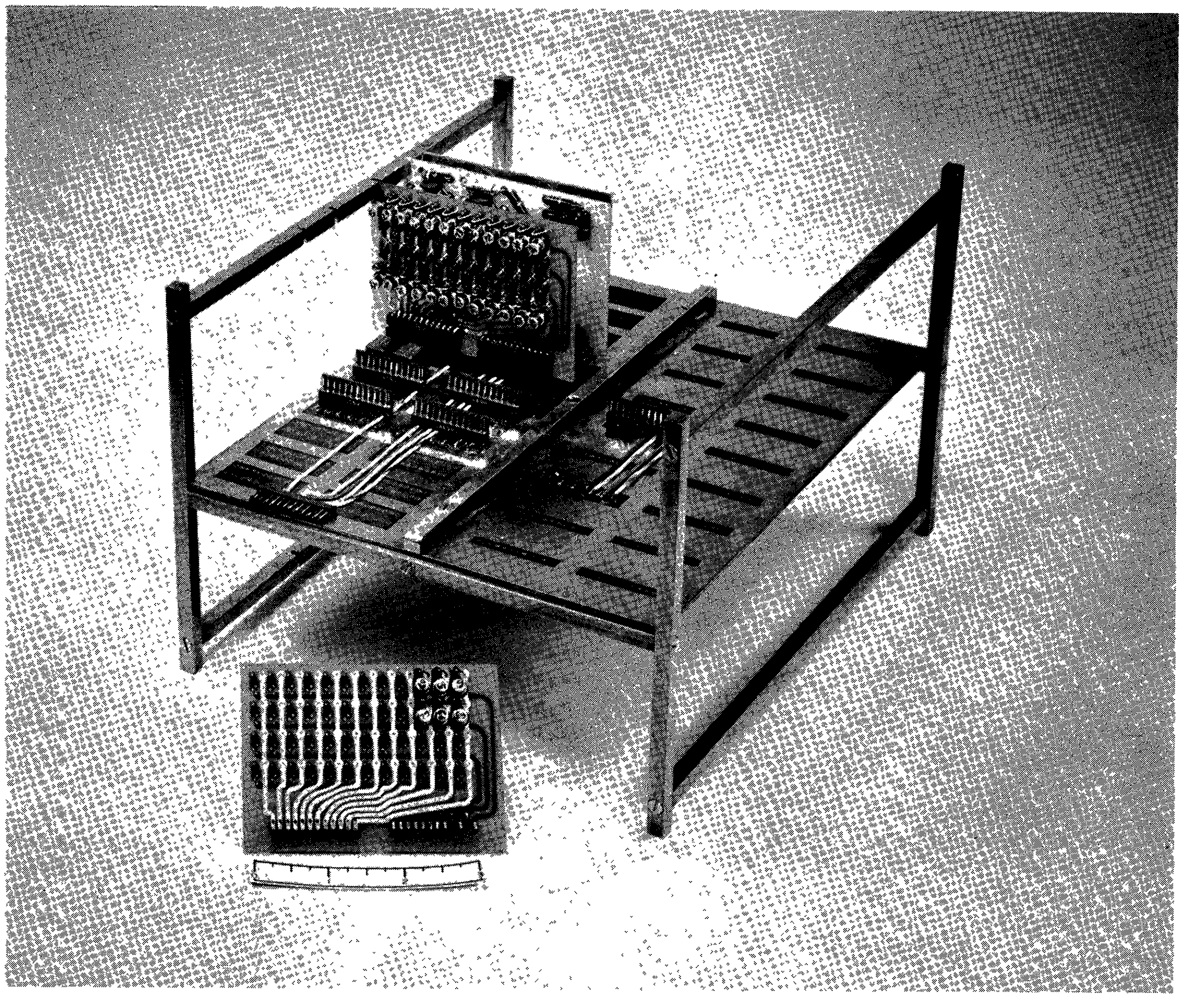


Figure 14. MEMORY TEST MODEL

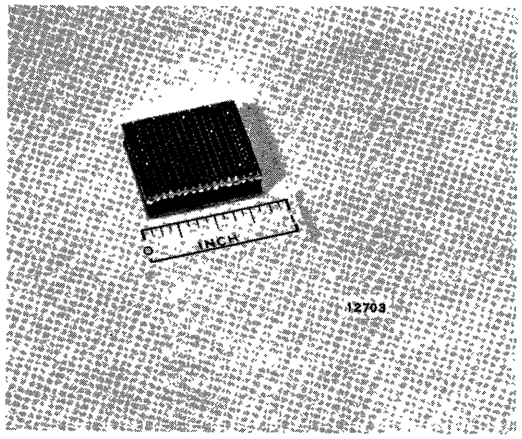


Figure 15. POSSIBLE MEMORY MATRIX CONFIGURATION

THE DEVELOPMENT OF A MULTIAPERTURE RELUCTANCE SWITCH

by

A. W. Vinal
International Business Machines Corporation
Space Guidance Center
Owego, New York

This paper describes the development of a substantially improved transfluxor type device called the Multiple Apertured Reluctance Switch (MARS). The MARS is similar to the classic transfluxor in that it comprises square loop magnetic material embodying two apertures. However, in deducing the details of a subtle switching phenomena, a powerful electrical control parameter was discovered, permitting geomechanic design of MARS devices mechanically characterized by two apertures having substantially the same inner perimeter (figure 1). Eliminating the necessity for the large aperture, a mechanical characteristic of the classic transfluxor, was necessary before a practical three-dimensional coincident current, non-destructive readout memory could be developed.

Basic Operation of MARS

Figure 2 illustrates a piece of square loop, square knee ferrite material containing two apertures having essentially the same inner perimeter. The left aperture has been identified as the read aperture and the right as the control aperture.

Figure 2 also shows two conductors passing through the read aperture. One is called a sense winding, labeled S1. It will sense, during read time, an electrical signal proportional to the time rate of change of the flux (ψ), ($d\psi/dt$) irreversibly switched about the read aperture. This flux reversal is caused by the current pulses interrogating the second conductor, A. The nature of the flux distributions, to be described later, determines whether a large or small sense signal, corresponding to a binary "one" or "zero" respectively, appears across the terminals of the sense winding.

The address conductor (B) passes through the control aperture. Current pulses applied to this conductor determine by means of flux distributions whether the "one" or "zero" information state occurs.

To qualitatively describe the operation of the MARS, a timing diagram is shown in figure 3. This diagram describes the time relationship between the read and control pulses applied to conductors A and B respectively. These pulses have been numbered to provide a simple method of distinguishing one pulse from another during the following discussion.

Figures 4a through 4d describe the saturated flux distribution that encircles the read aperture resulting from driving the current pulses (figure 4e) through it. Figure 4f describes the output signal present at the sense winding terminals during the application of corresponding current pulses No. 1 through No. 4.

In all flux distributions drawings shown, the current pulses applied to line A switch only the direction of flux encircling the read aperture, i.e., from a clockwise to a counterclockwise direction. A voltage pulse developed at the sense winding terminals corresponds to the time rate-of-change of this flux reversal. The magnitude of this voltage is proportional to:

$$e = -\frac{\Delta\psi}{\Delta T} = -As\,\frac{(Br + Bm)}{\Delta T} \qquad (1)$$

where

As = equivalent cross-sectional area of saturated flux encircling the read aperture

ΔT= time increment

Equation (1) describes the output voltage for the case where the flux encircling the read aperture is switched from $\pm$Br (the remnant flux density) to $\mp$Bm (maximum flux density).

Figure 5 shows an effective hysteresis curve that defines the flux density and magnetizing force relationship while the MARS is storing a binary "one", as viewed from the read aperture.

The amplitude of read current pulses No. 1 through No. 10 (figure 3), excluding control pulses No. 5 and No. 10, are of sufficient amplitude to saturate flux around the read aperture to a radius slightly less than its diameter.

Pulses No. 5 and No. 10 (figure 3), which interrogate the control aperture, are called the write "zero" and write "one" pulses respectively. The polarity of pulse No. 5 is not arbitrary. Its polarity must be in a direction that will produce reminant flux around the control aperture in the same direction last developed around the read aperture. For example, figure 4d shows flux in a clockwise direction established around the read aperture owing to the application of pulse No. 4 to line A. To establish the binary "zero" saturation flux distribution, current pulse No. 5 must be in a direction that establishes saturation flux in a clockwise direction around the control aperture.

Figure 6 shows the reminant flux distribution for the binary "zero" condition. Because of the shape, it is called the "pulley" flux pattern. This situation is analogous to the block condition of the transfluxor.[1] The belt of saturation flux encircling both read and control apertures formerly encircled only the read aperture. Specifically, this flux is near the control aperture, riding on the outer boundary of saturated flux encircling the control aperture. In essence, the application of the write "zero" control pulse switched the reluctance of the MARS to a higher value.

In figure 7 the solid base line passing through $-B_R$ is the higher reluctance or binary "zero" response excitation characteristic, as viewed from the read aperture. Current pulses that interrogate conductor A, formerly sufficient to switch flux encircling the read aperture, are now insufficient to switch the flux belt encircling both read and control apertures. When the pulley pattern exists, the sense winding signal at the read aperture is substantially zero. Bi-polar read pulses may be applied indefinitely to the A conductor with no effect upon the pulley pattern, if the pulses are below a critical amplitude (I_{RD} in figure 7). This critical amplitude (read destructivity threshold) and the switching mechanism is discussed later. The amplitude of the write "zero" current pulses (pulse No. 5) should be of sufficient magnitude to extend the flux in a radial direction from the center of the control aperture until the flux is tangent to the inside diameter of the read aperture. This is the situation shown in figure 6.

Unlike transfluxoroperation, the power and energy required to operate the MARS during the control phase is a minimum. This minimum energy requirement, in part, is accomplished by referencing the polarity of the write "zero" control pulse to that of the last read pulse. This technique eliminates the necessity for the control pulse to resaturate leg 1 of figure 8 since this leg was previously saturated during read time. (Leg 1 is the outer leg adjacent to the read aperture.) Re-saturating leg 1 during the writing of a binary "zero" is, therefore, useless and power consuming. However, this referencing technique requires the polarity of the write "zero" control pulse to be in a direction that will develop flux about the control aperture in the same direction as that last established around the read aperture. For example, in figure 8 the last read pulse No. 4, applied prior to writing a zero, establishes clockwise reminant flux about the read aperture. Similarly, control current pulse No. 5 (figure 3) applied to conductor B establishes clockwise saturation flux around the control aperture. This flux links leg No. 2 (figure 8) and opposes that portion of the flux encircling the read aperture. The action of the interference in leg 2 during the control pulse produces the "pulley" flux pattern shown in figure 6.

Figure 9 shows the flux distribution corresponding to the low reluctance state. This state is analogous to the unblocked condition of the transfluxor.[1] The flux belt previously encompassing both apertures (figure 6) is now encircling only the read aperture. This process by which the MARS is switched to this low reluctance state starts with the application of the write "one" control pulse No. 10 (figure 3).

This unblocked or binary "one" information state allows read current pulses to switch the direction of the saturation flux encircling the read aperture in the same manner described previously in reference to figure 4. This flux, may be alternately reversed in direction indefinitely. The only restriction is that the correct polarity be established around the read aperture prior to writing a "zero".

Topography Investigation

Discs of ferrite material prepared for experimental evaluation possessed magnetic properties exhibiting the rectangular B-H characteristics shown in figure 10. Each disc, pressed and appropriately sintered, was cylindrical, with diameter and thickness of 0.25 inches and 0.025 inches respectively. Two apertures were then ultrasonically drilled in each disc. The relative position and diameter of these apertures are listed in Table I. These freshly cut samples were re-sintered for a period of 10 minutes to relieve any abnormal strains that might have resulted from the drilling process. Mechanical characteristics and dimensions of these test samples are shown in figure 11.

Five basic experiment measurements were performed on each sample. The sequence of these experiments, listed below, was essential to the success of this development effort:

1) Test Sample - material homogeneity measurements.
2) Read destructivity threshold measurements.
3) Write "one" control properties
4) Write "zero" control properties
5) Write "one" - write "zero" switch time measurements.

The meaning of these experimental measurements relative to device operation are discussed later with the results of each experiment along with graphic plots of the appropriate test data. All experiments employed pulse techniques primarily because the mechanisms sought were switching phenomenon. If control of any of these mechanisms were to be developed, the transient nature of coherent energy (flux) interactions during external excitation would be the principal objectives of the experimental work.

Figure 12 shows the current pulse program used throughout the experiments. Each pulse in the sequence is numbered to simplify identification of individual pulses discussed.

Table I

Test Sample Dimensions in Inches

Sample No.	Ds	a	L	R_1	R_2	D
1	.003	.008	.050	.0125	.0125	0.250
2	.0365	.0115	.050	.0125	.0125	0.250
3	.039	.014	.050	.0125	.0125	0.250
4	.043	.018	.050	.0125	.0125	0.250
5	.045	.020	.050	.0125	.0125	0.250
6	.049	.024	.050	.0125	.0125	0.250
7	.055	.030	.050	.0125	.0125	0.250
8	.060	.035	.050	.0125	.0125	0.250

Each test sample was wired with No. 35 Formvar insulated wire to conform with the arrangement shown in figure 13.

In figure 13 sense winding S1 and S2 linking leg 1 and leg 3 were provided to measure the change in the effective vector magnitude of the flux density occuring within the magnetic material comprising leg 1 and leg 3 respectively. (The nature of the flux reversals occurring within these boundaries is discussed later.)

The following paragraphs describe the experimental procedure for obtaining the read and control response excitation characteristics of each test sample in order to determine the influence of aperture topography.

Uniformity Measurements

The first experiment determined electrical uniformity of the magnetic material comprising the test devices, both individually and as a group.

The amplitudes of positive and negative current pulses, such as pulses 1 and 2 in figure 12, applied to the read aperture of each test sample via conductor A, were gradually increased and maintained equal in magnitudes. The corresponding peak amplitude, switch, and peak time of the electrical signal sensed by winding S1 were recorded. Similar procedures were employed for each sample by sensing, with winding S2, irreversible flux switched about the control aperture owing to the bi-polar pulses passed through conductor B. This data was plotted for each sample, as exemplified by figure 14. The inner wall irreversible switching threshold for each sample was obtained by projecting the linear portion of the output pulse amplitude until it intersected the abcissa or current axis. It was important for subsequent tests to know that the switching threshold (I_O) for each sample had substantially the same value.

Write "One" Control Properties

The write one control properties for each test sample were obtained by using the following experimental techniques:

In figure 12 the magnitude of pulse No. 10, the write "one" control pulse interrogating the control aperture, was adjusted from an initial value of zero in increments of increasing amplitude. For each increment, the magnitude of current pulse No. 10 was recorded along with the corresponding peak amplitude, peak, and switch time of the read signal sensed by winding S1 during the application of read pulse No. 1. This response data could also have been taken during the first positive read pulse (pulse No. 11) occurring immediately after the application of the write "one" control pulse. However, there was a slight difference in the shape of the first readout signal obtained during pulse No. 11 compared to that measured at any other appropriate read time. But the area under each output response was the same. The difference in the observed response is believed to be associated with a slight disturbance in the distribution of flux encircling the read aperture caused by the application of the first positive read pulse following control pulse No. 10.

The measured response excitation (write "one" and "zero") characteristics for samples No. 2, No. 6, and No. 8 have been plotted collectively in figure 15. Two critical points (I_{RDS} and I_{RDF}) in the write "one" control property have been indicated for each sample.

I_{RDS} corresponds to the control current amplitude where the reluctance of the MARS, viewed from the read aperture, begins to switch from the <u>high</u> to the <u>low</u> reluctance state. I_{RDF} corresponds to the control current amplitude which is sufficient to completely switch the MARS into the <u>low</u> reluctance state. Flux distributions within the MARS device for the low and high reluctance states are shown in figures 9 and 6 respectively.

Before describing the significance of the write "one" control properties through an interpretation of the data, the procedure for obtaining the write "zero" control properties will be described.

Write "Zero" Control Characteristics

The order of the following control properties is essential for proper evaluation of the MARS device. The write "one" control properties must be established first because there is a third break current associated with this control process. It is called the <u>reflex</u> break current, I_{RB}.

The write "zero" control functions of the test samples were obtained by using the current pulse program shown in figure 12. For each sample under test, the amplitude of current pulse No. 10, (Write "one" control pulse) was adjusted to be in excess of the break current (I_{RDF}), but less than the reflex break current (I_{RB}). Once this condition was satisfied, current pulse No. 3, (figure 12) was adjusted from an initial value of zero in equal increasing increments. By recording the magnitude of current pulse No. 3 at each increment and the associated peak amplitude, peak, and switch time of the signal appearing at the terminals of winding S1 during read current pulse No. 5, figure 12, we obtained the write "zero" response excitation characteristics plotted in figure 15.

Unlike the write "one" control function, there are only two current break points indicated in the figures, I_{RIS} and I_{RIF}. I_{RIS} corresponds to the current magnitude where the reluctance of the MARS starts to increase (RIS - Reluctance Increase Start). I_{RIF} corresponds to the current amplitude of the write "zero" control pulse where the reluctance of the MARS is completely switched to the high reluctance state, (I_{RIF} - Reluctance Increase Finish).

An interesting phenomenon, shown in figure 15; is associated with the write "one" control response excitation characteristics. For each sample tested, break point (I_{RDS}) was substantially identical. The results of this experiment indicated that:

$\underline{I_{RDS}}$ <u>is independent of the separation distance between the read and control apertures, and it is directly proportional to the inner perimeter of the control aperture and the switching coercivity of the magnetic material comprising the device.</u>

Break points I_{RIS}, I_{RIF}, and I_{RDF}, however, are substantially effected by aperture separation.

For a MARS device to be conveniently operated in a three dimensional, coincedent current system, the following are critical design criteria:

$$I_{RDS} = I_{RIS} \tag{1}$$

$$I_{RDS} = 66\% I_{RDF} \tag{2}$$

$$I_{RIS} = 66\% I_{RIF} \tag{3}$$

Figure 15 shows that an increase in aperture separation increases the current amplitude at which thresholds I_{RIS}, I_{RIF}, and I_{RDF} occur. For wide aperture separation distances, exemplified by samples 6 and 8, two criteria specified by equations (1) and (3) are violated. If this situation were tolerated, considerable power would be required to operate the device. Fortunately, equation (1) and equation (3) are compatible with low-power operation since they require close aperture proximity. However, a later section describes a phenomenon, the read destructivity threshold, which if uncontrolled prohibits close aperture proximity. The solution to this dilemma is discussed later and a unique method for electrically controlling this destructivity threshold is described.

Inner Wall Reflex Switching

Read Destructivity Threshold

The read destructivity threshold is the amplitude at which read current pulses start to reduce the reluctance of the MARS device, viewed from the read aperture. This phenomon was referred to as spurious unblocking in reference 1. High and low reluctance states define the binary "zero" and "one" information states respectively. (Reluctance as used in this paper defines the relative degree of impedance encountered at the read aperture by the energizing means to switching irreversibly the flux in leg 1 (figure 13) of the MARS).

It is essential to understand the destructivity threshold in terms of flux distributions and particularly the switching mechanisms responsible for it. Characteristically, the existence of this phenomena determines to a great extent the limitations of multiapertured devices for practical applications in random access, non-destructive memory systems. Furthermore, if these subtle switching phenomena are understood, the optimum geomechanics of this and other multipath devices can be determined both experimentally and analytically.

The physical arrangement of the conductors passing through each experimental test sample is shown in figure 13. Conductors A and B permit external energizing means to influence the material energy state in the vicinity of the read and control apertures. Conductors S1 and S2 sense the change occurring within the material bounds comprising leg-1 and leg-3 respectively. Regarding current polarities, the reference direction adopted in this paper corresponds to applying positive current pulses to conductors A and B in the direction indicated by arrows in figure 13 to produce counterclockwise energization around the respective apertures.

Read destructivity thresholds for each test sample were determined experimentally by applying the current pulse program shown in figure 16.

Figure 17 is a plot of the switch time of the "zero" readout signal versus the amplitude of read pulse 6 and 7. This data was obtained by recording the electrical properties of the "zero" readout signal occurring during read pulse No. 9.

The readout signal occurring at pulse time No. 2, called a reference signal, was also observed, but only as a relative measure of the degree of destructivity being induced by pulse No. 6. This reference signal displayed the electrical output associated with each test sample after it was fully switched by control pulse No. 10 to the binary "one" information state. Referring to figure 17, note that for the reference polarity established, no destructivity threshold occurred for clockwise flux inducing read current pulses.

It is important to know the amplitude at which positive read current pulses start partial destruction of the binary zero information state, and how the geomechanic properties of the test samples influence this phenomenon.

Figure 16 shows the pulse program used to measure the positive and negative read destructivity threshold by overdriving read pulse No. 6 and 7 respectively.

Figure 18 shows the effect the width of the common leg, separating the read and control apertures, on the current amplitude at which the destructivity threshold occurred.

For all samples tested, the amplitude of the positive read current pulse, corresponding to the destructivity threshold, was in excess of the magnetizing force I_O required to just irreversibly switch flux in the unblocked wall of the read aperture, but substantially less than twice this magnitude. This situation is shown in figure 17 where the "zero" readout response excitation of sample No. 2 and No. 8 are plotted collectively. Notice that the destructivity threshold I_{RD} is substantially less than twice I_O. This situation apparently compromises the effective use of multipath devices with similar apertures in a coincident current, three dimensional matrices. The nature of this threshold and its severe compromises provided a motivation to deduce, by experimental techniques, the mechanism responsible for it, and hopefully to evolve a suitable control technique. The material presented subsequently describes experimental techniques which led to the discovery of the switching phenomenon identified as "inner wall reflex switching". These critical experiments, conceived to verify this switching mechanism, provided a powerful control technique and design variable.

Prior to discovering the actual nature of this switching phenomenon, geomechanical techniques were used for increasing the relative current level at which the destructivity threshold occurs. These techniques are exemplified by the experimental characteristics plotted in figure 18 and by the classic transfluxor where the inner perimeter of the control aperture is substantially larger than that perimeter comprising the read aperture.[1, 2]

The switching mechanism associated with the read destructivity threshold is shown in figures 19a through 19d. These drawings show the distribution of flux thought to exist in the proximity of the read and control apertures during different stages of deformation. The degree of deformation is proportional to the amplitude of positive current pulses applied to the read aperture. Referring to figure 19a, leg 2 separating the read and control apertures was saturated in the reference direction (up) during the generation of the pulley flux pattern by control pulse No. 3 (figure 16). The flux density (B_r) in this leg can be increased, depending on material rectangularity, by externally applying to the read aperture a magnetizing force in only one direction. For the established reference direction, this corresponds to the positive current pulses shown in figure 16. The intensity of the $\overline{H}$ field propagated from the conductor can be calculated to be proportional to the current and inversely proportional to the radial distance from the conductor. The action of the $\overline{H}$ field propagated from conductor A increases the flux density in leg 2 in diminishing proportions depending upon the radial distance of leg 2 from the conductor and simultaneously diminishes flux density in leg 3 by the action of the same vortex source, particularly in the vicinity of the wall area of the control aperture.

As the amplitude of the positive read current is increased, the coherent energy (flux) distribution internal to the magnetic material at the wall of the control aperture and tangent to leg 3 reaches a critical level. When this critical level is surpassed, the coherent energy directed clockwise around the wall of the control aperture reflexes back tangent to the remote side of the control aperture. This switching phenomenon, shown in figure 19d, has been called "inner-wall reflex switching".

The necessary conditions believed to be required to perpetrate this reflex switching is summarized by the following hypothesis, which is subsequently confirmed by experimental evidence.

A given region in bounded magnetic material that is suitably influenced by energy generated externally contains adequate internal coherent energy (including that produced by the external influence) to cause a reversal in the direction of coherent energy in another region. This second region undergoing reversal is in the immediate proximity of the first, and it has a lesser mean path length.

When this inner wall reflex switching is allowed to occur, it acts as a gate permitting flux to irreversibly switch in a counterclockwise direction concentrically about the read aperture. This situation is shown in figure 19d. Furthermore, the amount of flux permitted to switch about the read aperture is directly proportional to that amount reflexed about the wall of the control aperture. This, therefore, is the mechanism alluded to earlier as being responsible for the read destructivity threshold. Although the occurrence of this switching mechanism is undesirable in memory device applications, it undoubtedly is useful for others.

To experimentally determine that reflex switching is the mechanism responsible for the read destructivity threshold, and that it obeys the hypothesis outlined, some critical experiments were performed. Specifically, these experiments were designed to determine the following:

1) That reflex switching occurs.

2) That it occurs in the wall of the control aperture.

3) That by occurring first, it acts like a proportional flux gate producing the effect at the read aperture just described.

Does reflex switching occur? Conceptually, the experiment designed to answer this question is simple: By applying the train of current pulses shown in figure 16 to the appropriate apertures, and then increasing the amplitude of current pulse No. 6 in excess of the destructivity threshold, a signal should appear at the terminals of sense winding S2 (figure 13). The anticipated signal was observed. This would indicate that a change in the vector magnitude of the total coherent energy in leg 3 had actually occurred. As expected, the switching characteristics of the electrical signal detected were considerably different from those normally associated with irreversibly switching a similar amount of flux in a path closed about an external vortex source. The principal difference observed was a characteristically long trailing tail on the detected electrical response signals. Figure 20 depicts the electrical signal observed compared to the switching that encloses an external vortex source.

It was more difficult to devise an experiment for determining whether reflex switching instigates the destructivity threshold by occurring first and at the wall of the control aperture.

The experiments conducted show that the results anticipated occurred by using the pulse program shown in figure 21 and proceeding with the destructivity threshold measurements in exactly the same sequence already described to obtain the plot shown in figure 17. The salient results are shown in figure 22 which plots the switch time for the "zero" read out signal as a function of positive read current pulse amplitudes and the amplitude of pulse No. 13. The magnitude of pulse No. 13 is shown as the running parameter in figure 22. We see that bias pulse No. 13 substantially influenced the current amplitude I_{RD} at which positive read current pulses initiate destruction of the binary "zero" information state.

The experiment just described was carried further by determining the relationship of the destructivity threshold I_{RD} as a function of the amplitude of pulse No. 13, defined as the "inner wall bias pulse". The results of this latter experiment are plotted in figure 23.

The data plotted in figure 22 indicates that the energy propagated by the conductor carrying bias pulse No. 13 applied under the conditions specified by the experiment retarded reflex switching. Evidence of this fact is shown in figure 22 where the read destructivity current threshold (I_{RD}) is increased by an amount proportional to the magnitude of bias pulse 13. Figure 23 further exemplifies this condition by showing that the read destructivity threshold is effectually controlled by an amount substantially equal to the magnitude of bias pulse 13, within the limits indicated. The shape of the curve in figure 23 leads one to conclude that reflex switching initially occurs in the wall of the control aperture and not elsewhere.

If reflex switching did not occur in the wall of the control aperture, but in some other more remote position in leg 3 (figure 19d), the control provided by the bias pulse shown in figure 23 would not break down at the current level indicated. Specifically, figure 23 shows that the current level (I_o) corresponding to loss of control by the bias pulse is the current amplitude previously determined to be equal to the irreversible switching threshold of the wall of the control aperture.

If reflex switching had started at a position more remote than in the wall of the control aperture, the bias control function shown in figure 23 would have broken at some correspondingly higher amplitude.

Turning to the results of the first experiment, a signal was detected by sense winding No. 2 during an overdriven positive read pulse. Specifically, the electrical characteristics of this reflex signal (figure 21) were different from those normally encountered by irreversible switching of coherent energy about a path enclosing an external vortex source. If reflex switching at the wall of the control aperture acts as a gate, permitting coherent energy to be irreversibly switched around the read aperture enclosing the vortex source, the electrical signal sensed at winding S1 should have indentically the same electrical characteristics as that simultaneously sensed by winding S2. Comparing these experimentally measured signals revealed no detectable difference in their response characteristics; also, both signals possessed the response characteristics peculiar to reflex switching. The fact that inner wall reflex switching occurs first (acting as a proportional flux gate) as described permitted controlling the read destructivity threshold by biasing the control aperture.

The effectiveness of this bias technique is illustrated in figures 24a through 24c, using test sample No. 3 for the conditions indicated below. These figures are exact enlarged copies of the actual oscilloscope traces.

Figure 24a shows nondestructive read of "zero" and "one" with read currents of 300 ma full-select, below read destructivity threshold and where no inner wall bias was used.

I_{Read} = ±300 ma full-select

$I_{WR``0"}$ = 600 ma full-select

$I_{WR``1"}$ = 450 ma full-select

Figure 24b shows nondestructive read of "zero" and "one" with read currents of 400 ma full-select. Note that the positive read current is above read destructivity threshold, hence the increase switching time of the "zero" readout. In figure 24b:

I_{Read} = ±400 ma full-select

$I_{WR``0"}$ = 600 ma full-select

$I_{WR``1"}$ = 450 ma full-select

Figure 24c shows nondestructive read of "zero" and "one" with drive conditions as shown in figure 24b. Inner wall bias of 100 ma has been used to shift read destructivity threshold. Note the reduction of peak amplitude and switch time of "zero" signal.

The experiments outlined in this section indicate the following:

1) Inner wall reflex switching occurs.

2) Inner wall reflex switching occurs in the wall of the control aperture.

3) By occurring first, inner wall reflex switching acts like a proportional flux gate, allowing switching to occur about the read aperture.

4) Inner wall reflex switching is the mechanism responsible for the read destructivity threshold.

Write "One" Break Point-I_{RDS}

From experimental work outlined earlier I_{RDS} was found to be:

1) Independent of aperture separation

2) Directly proportional to the inner perimeter of the control aperture.

3) Directly proportional to the inner wall switching coercivity of the magnetic material comprising the device.

These experimental results provide useful aperture and topography design parameters. The nature of the I_{RDS} properties and the mechanisms responsible need not be answered to use the experimental findings for engineering purposes.

The switching mechanism associated with the write "one" control process is illustrated by figures 25a through 25e. These drawings show the distribution of coherent energy (flux) in the proximity of the read and control apertures thought to exist during different stages of switching. The degree of switching indicated is proportional to the amplitude of the write "one" control pulse. Referring to figures 25a through 25d we can see that the effects of the write "one" control pulse instigates a complicated redistribution of coherent energy. This redistribution involves reflex switching, but not in either aperture wall. Reflex switching is believed to exist in the area shown in the figure because it was observed that part of the electrical signal detected by winding S2 (figure 13) during the write "one" control pulse exhibited the characteristics peculiar to reflex switching described previously.

In addition to the hypothesis cited, this reflex switching is believed to be instigated spontaneously by the action of the innermost band of coherent energy formerly encircling both apertures closing directly about the read aperture as shown in figure 25b. This closure can only be caused by a supplementary component of energy propagated internal to the magnetic material boundaries from conductor B, which carries the write "one" control current.

Reflex Break Point I_{RB}

Experimental work reported earlier indicated that a third break point exists in the write "one" response excitation characteristic. This break point, I_{RB} in figure 15, was referred to as the refle x break current. Increasing the amplitude of the write "one" control current beyond the reflux break point affects the readout signal subsequently sensed at the read aperture in essentially the same fashion as when writing a "zero". The mechanism responsible for this phenomenon is defined as inner-wall reflex switching. With the exception of a slight difference in the peripheral flux distribution, the mechanics of the reflex break phenomenon are identical to those described previously as responsible for the read destructivity threshold.

Most of the text regarding inner-wall reflex switching would apply to the reflex break phenomenon if the words "read" and "control" are interchanged.

Figure 25 shows the distribution of coherent energy in the proximity of the read and control apertures for the conditions indicated. A second region of reflex switching is shown in figure 25d, specifically in the wall behind the read aperture. To show that the reflex break phenomena involve inner-wall reflex switching as noted in figure 25e, a pulse bias technique was employed, similar to that disclosed earlier. Figure 26 is a diagram of the current pulse program used. As shown, bias pulse No. 11 and write "one" control pulse No. 8 were applied to their respective apertures simultaneously. The amplitude of the bias pulse was maintained below the inner-wall switching threshold of the read aperture. Experimental data plotted in figure 27 indicates that the reflex break threshold is increased by an amount proportional to the magnitude of bias pulse No. 11, within the limits indicated. The bias current loses control at the magnitude just capable of irreversibly switching the inner perimeter of the read-aperture. The control latitude offered by this bias pulse permits the operation of production MARS devices in a unique three dimensional matrix. In this matrix it is virtually impossible to surpass the reflex break point.

Experimental evaluation showed that I_{RDS}, I_{RDF}, I_{RIS}, and I_{RIF} are uneffected by the presence of bias current of either polarity, if magnitude is below that amount required to switch the wall of the aperture through which it is caused to pass.

Verification of the "Pulley" Flux Pattern

Reference has been made to the existence of the "pulley" flux pattern. Verification of its existence was deferred until now because proof is better understood after assimilating some of the concepts developed earlier.

The saturation flux drawn in figure 28a illustrates the coherent energy distribution in the proximity of the read aperture owing to current pulse No. 2 (figure 16). Current pulse No. 3 applied to conductor B is in a direction that establishes clockwise flux around the control aperture, as illustrated in figure 28b. This flux opposes that portion linking leg 2 already established around the read aperture. Under these circumstances, two alternative flux distributions might seem feasible as shown in figures 28c and 28d. Figure 28c is the familiar "pulley" pattern referred to frequently and figure 28d shows the alternate case where a form of reflex switching has occurred.

The purpose of the following experiments is to provide conclusive experimental evidence of the "pulley" pattern of figure 29. It appears from figure 28c that the generation of this flux distribution involves no net change in the flux linking leg 1. On the other hand, figure 29d illustrates a situation in which the total net vector flux linking leg 1 is numerically reduced to zero. If this later condition were the true resultant flux distribution instead of the "pulley" pattern, a substantial signal would be measurable across the terminals of winding S1 (figure 13) during control pulse No. 3. Experimental evidence does not substantiate the flux distribution of figure 29d since there was no switching signal present at the terminal of winding S1 during control pulse No. 3. This experiment indicates the existence of the "pulley" flux pattern (figure 29c) since its generation does not alter the resultant vector magnitude of flux already established in leg 1 by read pulse No. 4; therefore, no switching signal should be observed. Additional verification of the pulley flux distribution was provided by the electrical performance of the "tear drop" MARS device shown in figure 30. The external boundary of the "tear-drop" device was designed to confrom precisely with the outer periphery of the "pulley" flux distribution (figure 28c). The electrical performance of this device was identical to its counterpart in an unbounded form of a 1/4 inch disc with the same aperture topography.

Tangible Results

The more immediately tangible results of this study are the following:

1. key-hole MARS
2. three dimensional matrix array

These developments are discussed in the following paragraphs:

The Key Hole MARS

The salient characteristics of the keyhole device are:

1. Non-destructive read out.
2. Ideally suited for three dimensional, coincident current operation.
3. Half-select currents required for the read and control operations are the same magnitude.
4. Speed capabilities for both read and store modes are at least as fast as a three dimensional toroidal core memory.
5. Device size permits matrix densities of at least 3000 bits per cubic inch.
6. Mechanical characteristics afford relative ease of manufacture and handling.

Figure 30 is a schematic of the production key-hole MARS. The specific design criteria dominating the aperture topography were as follows:

$$I_{RDS} = I_{RIS} \qquad (1)$$

$$92\% I_{OR} < I_{RDS} \leq I_{OR} \qquad (2)$$

$$I_{RIS} = 68\% I_{RIF} \qquad (3)$$

$$I_{RDS} = 68\% I_{RDF} \qquad (4)$$

The name "key-hole" was chosen because of the suggestive shape, which was selected to simplify automatic testing, handling, and array assembly. The response excitation characteristics electrically defining the "key-hole" MARS are shown in figures 31a through 31c. The break points I_{RIS}, I_{RDS}, and I_{OR} were controlled through design of the aperture topography. Break points I_{RD} and I_{RB} are controlled by uniquely employing inner-wall bias as an integral part of the three dimensional MARS matrix array.

3-D MARS Matrix Address Configuration

Figures 32a through 32c show three coincident current selection schemes. The technique shown in figure 32a is the conventional method for instrumenting transfluxor type devices.[3] Figure 32b illustrates the effective 5-wire system (sense and inhibit not shown) conceived and designed as an integral part of the "key-hole" MARS. The vertical (X) coordinate address selection means "link both apertures (read and control) of each and every element comprising that selected column coordinate". This selection technique automatically provides the inner-wall bias necessary for proper operation of the "key-hole" MARS. Without inner-wall bias, the break points $I_{RB}{}^1$ and $I_{RD}{}^1$ occur at substantially lower excitation levels, as shown in figures 31a and 31c respectively. The ability to employ this matrix technique involves solving the geomechanic design criteria indicated in equation (1) and equation (2). Advantages of this technique are as follows:

1. Only one driving and selection means is required for column selection instead of two.
2. Transmission characteristics resemble those of an ideal transmission line.

The selection scheme shown in figure 33c is called the effective 4-wire, 3-D MARS array. As shown, the X coordinate selection technique is indentical to that described for figure 33b. However, the Y coordinate selection means is connected to half-select the read apertures of all devices comprising one row and the control apertures of all devices comprising an adjacent row.

Figure 34 is a photograph showing a section of a 4-wire MARS matrix. This technique, similar to that described for X coordinate selection, eliminates the need for two different selection and driving means. Similarly it provides ideal address transmission line characteristics. The 4-wire MARS matrix array permits a three dimensional Random Non-Destructive Advanced Memory (RANDAM) to be instrumented with no more selection or driving means required than those required to instrument a three dimensional, toroidal core matrix.

Summary

The results of this study provided useful understanding of subtle switching mechanisms peculiar to multi-aperture ferro magnetic devices. The experimental techniques employed to study these switching phenomenon provided a new and powerful control technique called inner wall pulse bias. This bias technique provided the suplimentary control variable which, when combined with geomechanic techniques, permitted the development of the Key Hole MARS and embodyment on a unique 3-D coincident current matrix array.

Acknowledgment

The author wishes to express his thanks to R. Betts, K. Clark, and M. Sulich whose work, in providing materials and devices to specifications, was essential to the success of this development. The author is indebted also to J. Coffin and J. Owen for their work on instrumentation and the gathering of experimental test data.

References

1. J. A. Rajchman and A. W. Lo, "The Transfluxor - A Magnetic Gate with Stored Variable Setting" - RCA Rev. Vol. 16 PP 303-311, June 1955.

2. J. A. Rajchman and A. W. Lo, "The Transfluxor" - Proc. IRE Vol. 44, PP 321-332 - March 1956.

3. J. A. Rajchman, "Computer Memories - A Survey of the State of the Art" - Proc. IRE, Vol. 49, No. 1, PP 104-127, January 1961.

Bibliography

1. D. A. Buck and W. I. Frank, "Non-destructive Sensing of Magnetic Cores" - Commun. and Electronics, No. 10(AIEE Trans. pt. 1, Vol. 72) PP 322-330, January 1954.

2. C. L. Wanlass and S. D. Wanlass, "BIAX High Speed Magnetic Computer Element", 1959 Wescon Convention Record Pt 4, PP 40, 54.

3. R. M. Tillman, "Fluxlock - a Non-destructive Random Access Electrically Alterable High Speed Memory Technique Using Standard Ferrite Memory Cores", IRE Transactions on Electronic Computers, Vol. EC-9 PP 323-328, September 1960.

4. R. Thorensen and W. R. Ansenault, "A New Non-destructive Read for Magnetic Cores ", Proc WJCC PP 111-116, March 1-3, 1955.

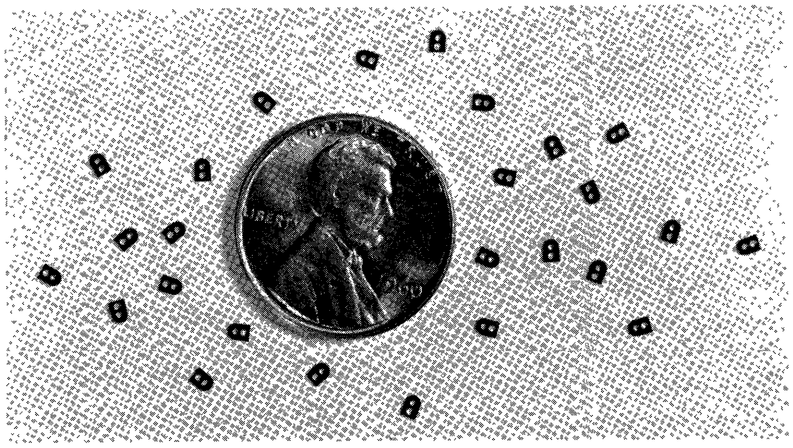


Figure 1. MULTIPLE APERTURED RELUCTANCE SWITCH (MARS)DEVICES (PHOTO)

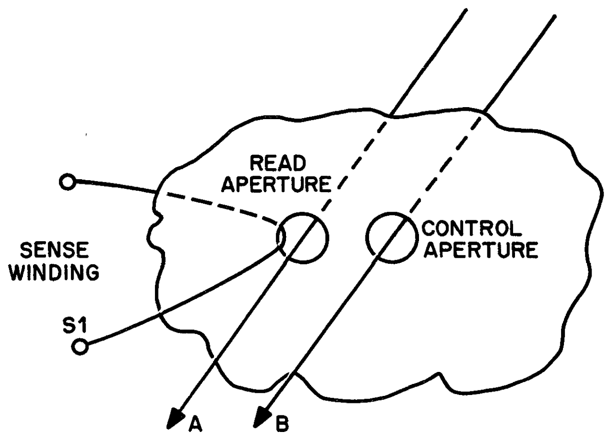


Figure 2. SCHEMATIC OF BASIC MARS DEVICE

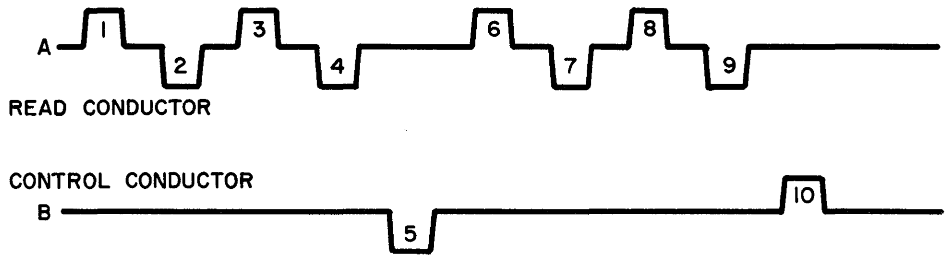


Figure 3. TIMING DIAGRAM

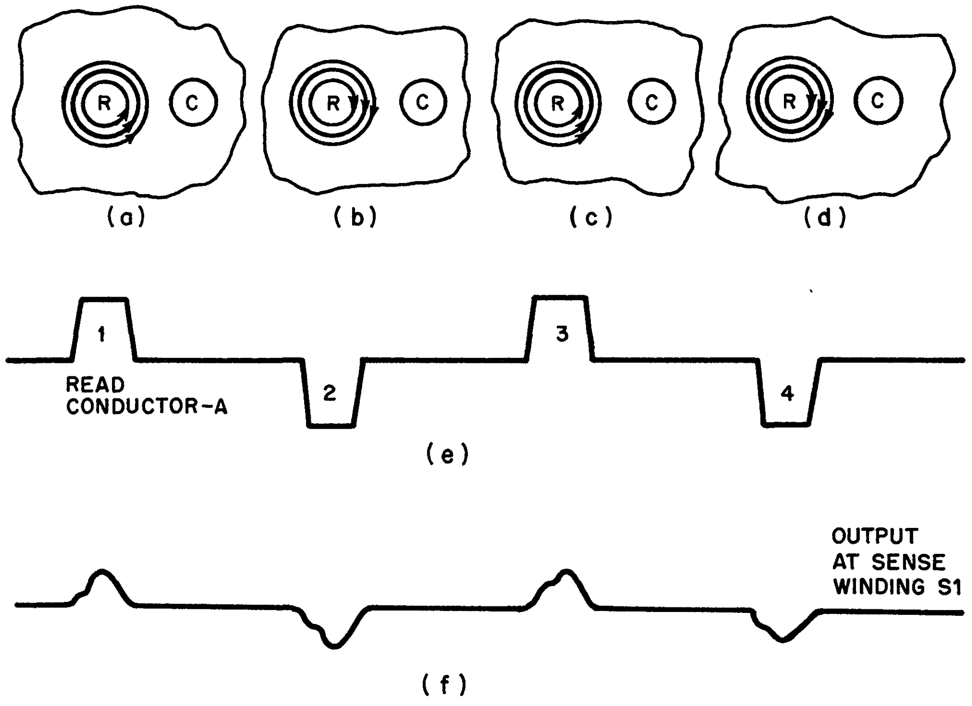


Figure 4. SATURATED FLUX DISTRIBUTION AND OUTPUT SIGNAL

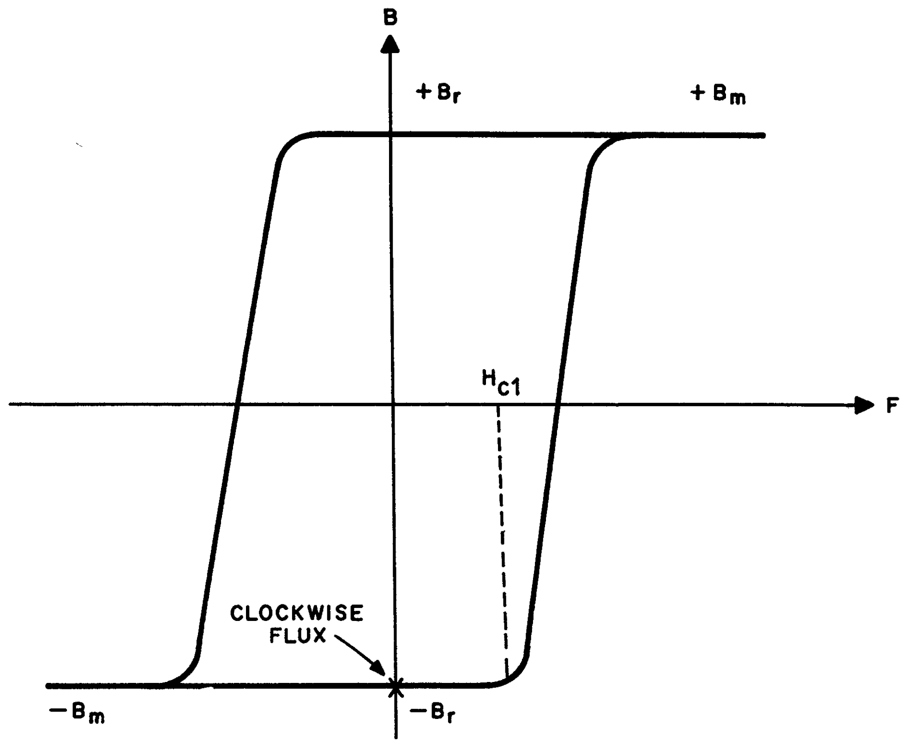


Figure 5. EFFECTIVE HYSTERESIS CURVE FOR MARS DEVICE STORING A BINARY "ONE"

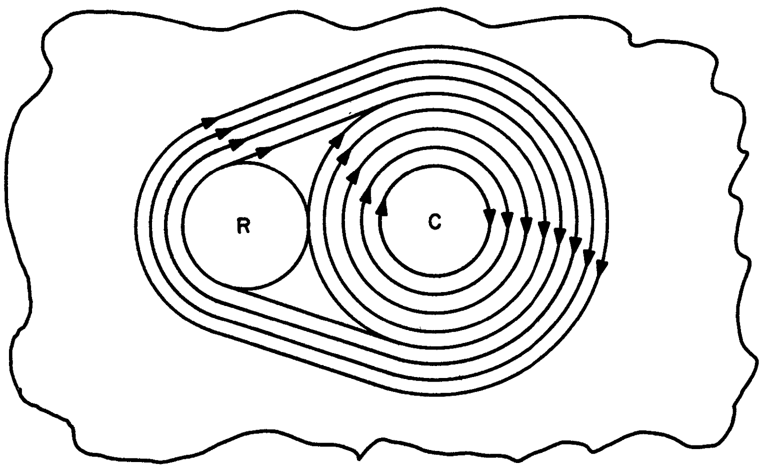


Figure 6. MARS DEVICE STORING "ZERO"

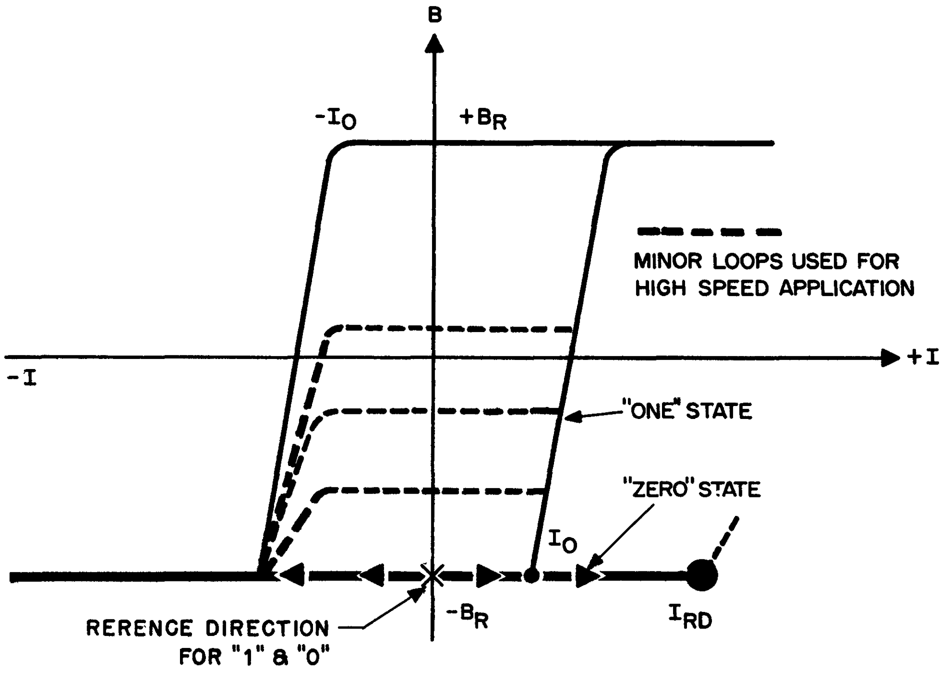


Figure 7. SUPERPOSITION OF RESPONSE EXCITATION CHARACTERISTICS OF BINARY "ONE" AND "ZERO" INFORMATION STATES

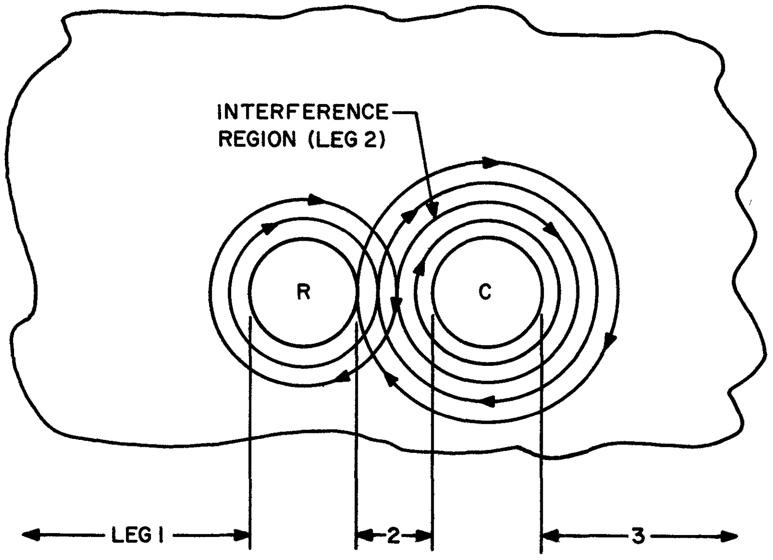


Figure 8. CLOCKWISE SATURATION FLUX-LAST READ PULSE APPLIED BEFORE WRITING "ZERO"

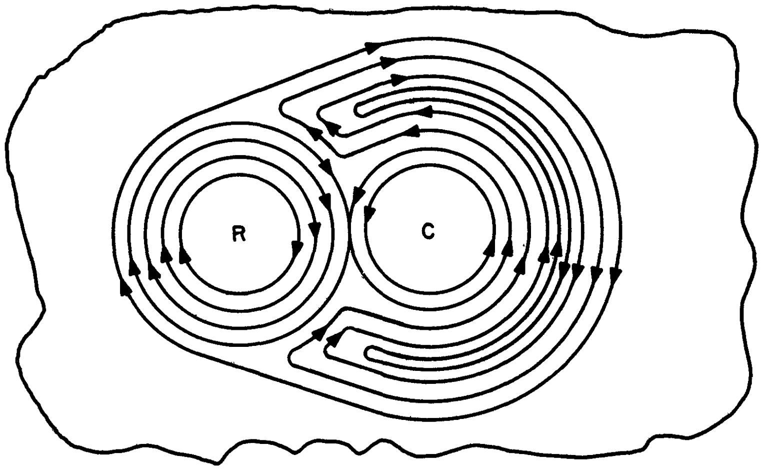


Figure 9.
FLUX DISTRIBUTION CORRESPONDING TO LOW RELUCTANCE STATE

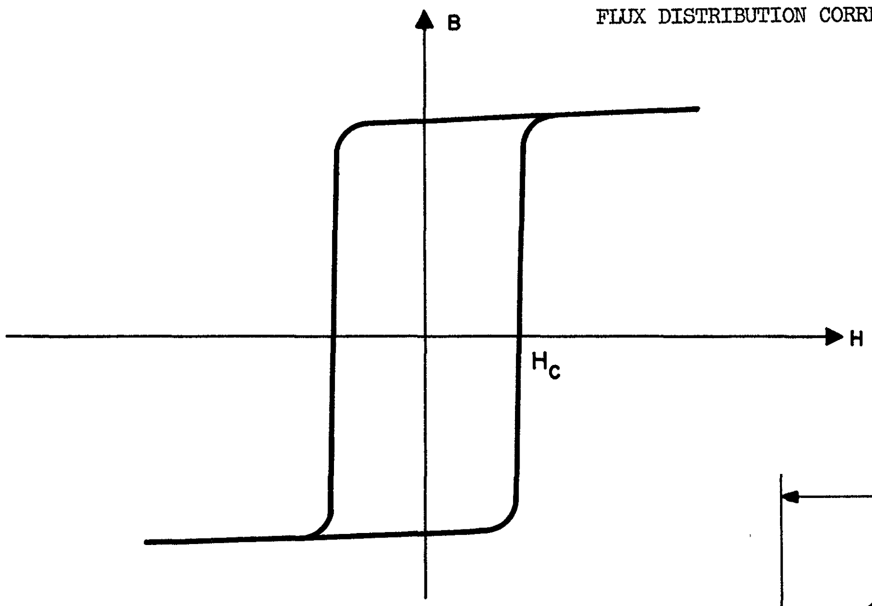


Figure 10. MATERIAL BH CHARACTERISTICS

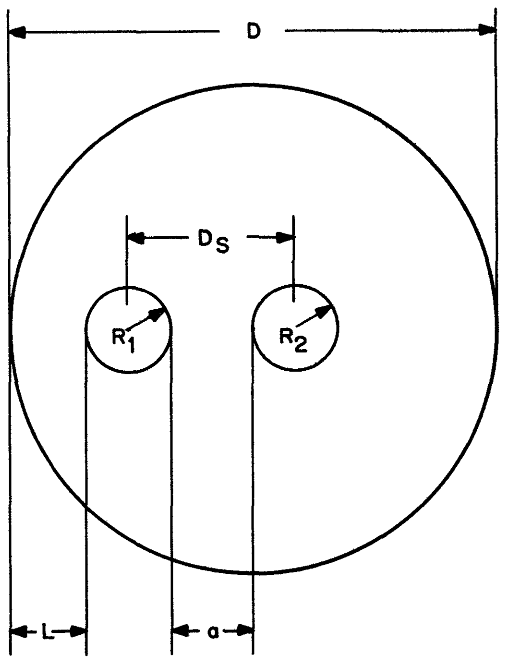


Figure 11. TEST SAMPLE APERTURE TOPOGRAPHY

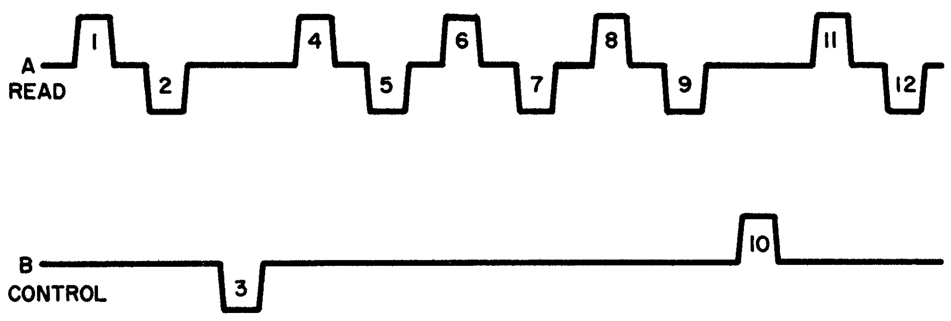


Figure 12. TEST CURRENT PULSE PROGRAM

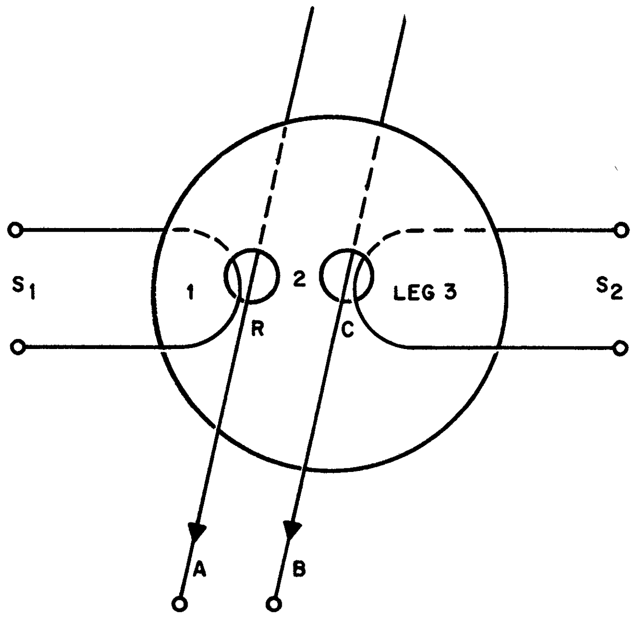


Figure 13. WIRING DIAGRAM OF TEST SAMPLES

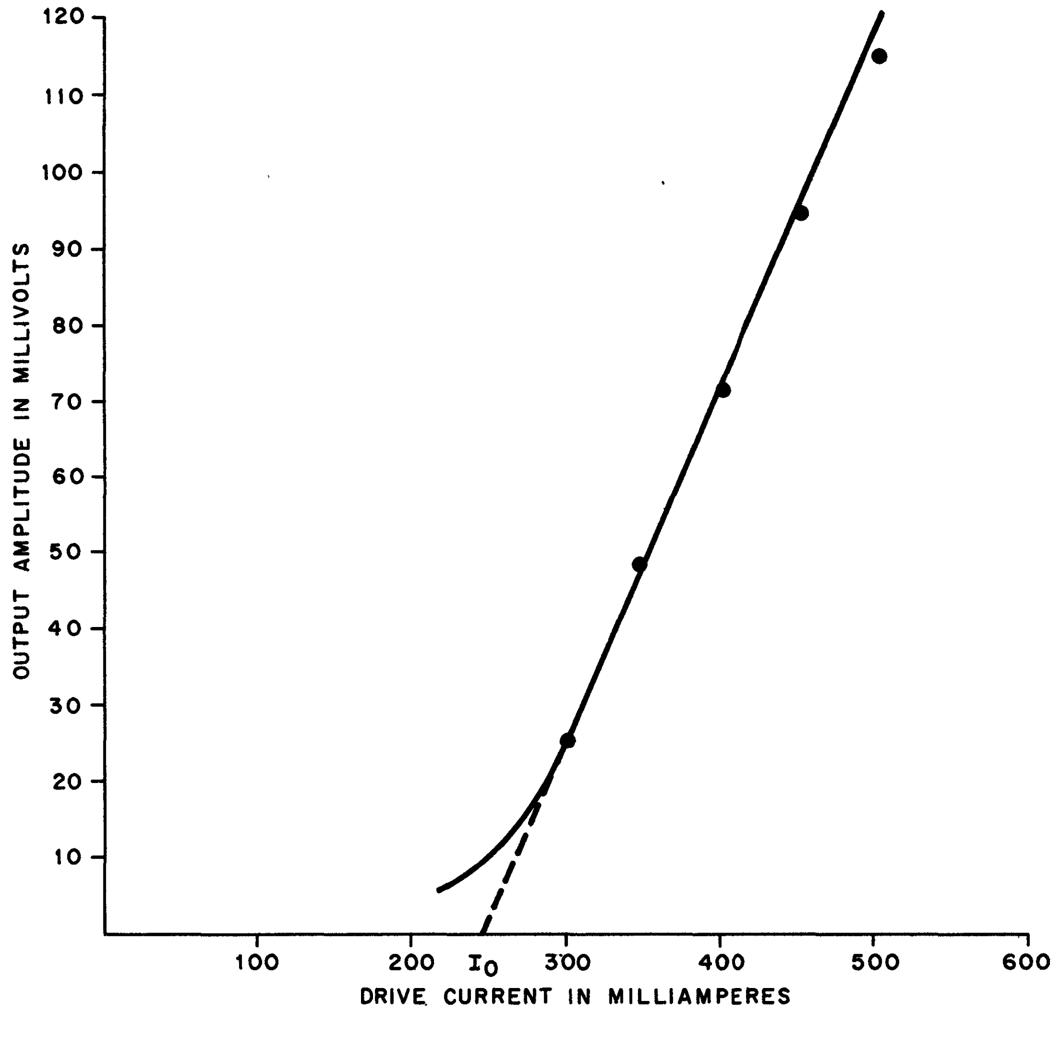


Figure 14. PLOT OF TYPICAL RESPONSE CHARACTERISTIC UNIFORMITY MEASUREMENTS

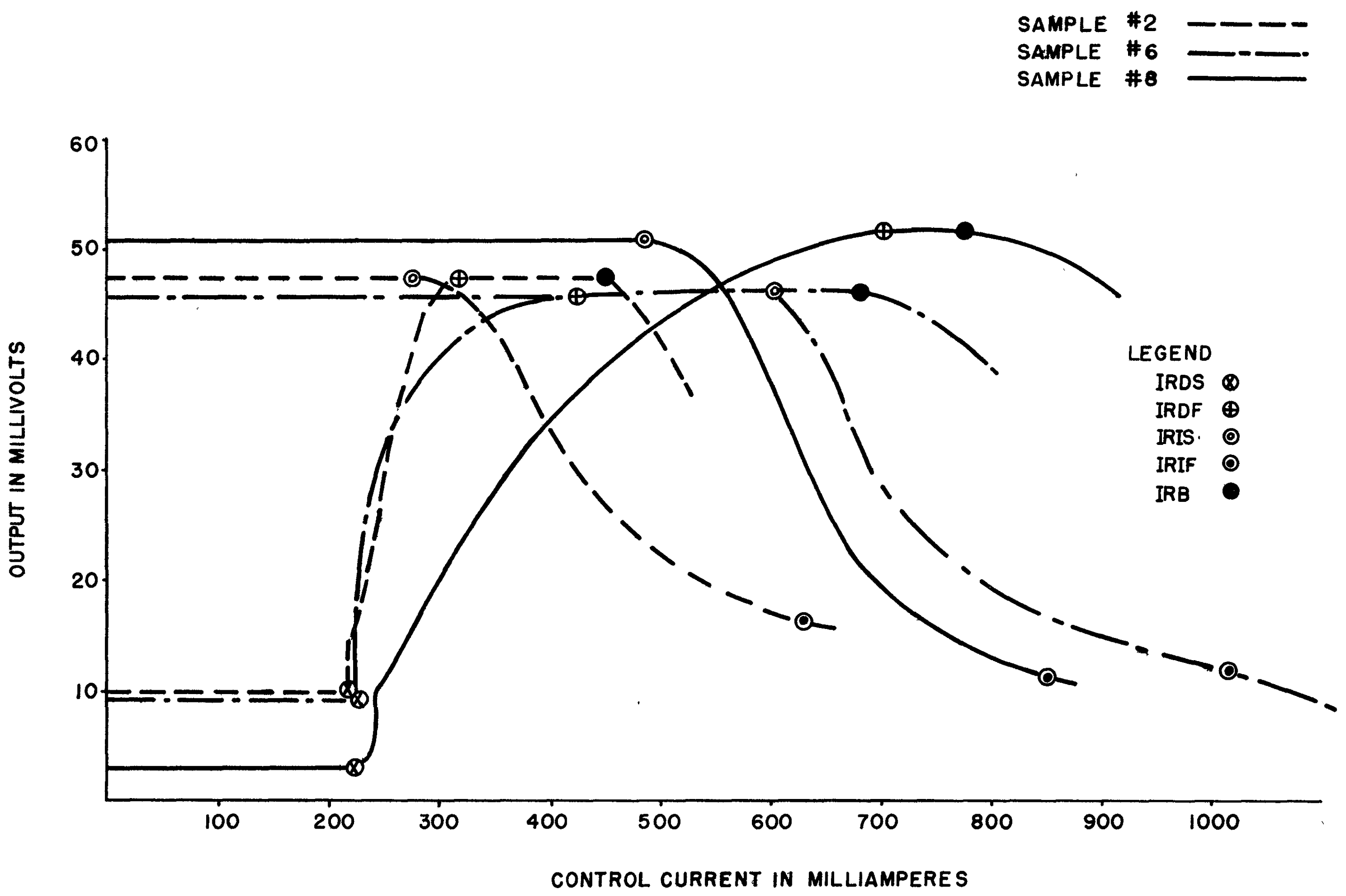


Figure 15. CONTROL RESPONSE EXCITATION CHARACTERISTICS

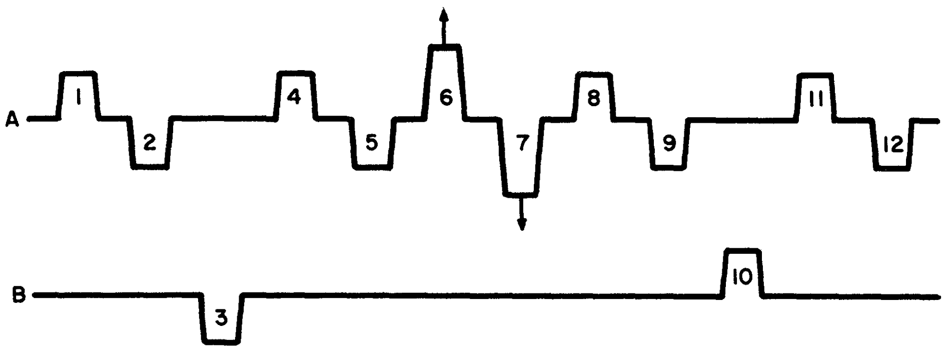


Figure 16. PULSE PROGRAM USED TO MEASURE READ DESTRUCTIVITY THRESHOLD

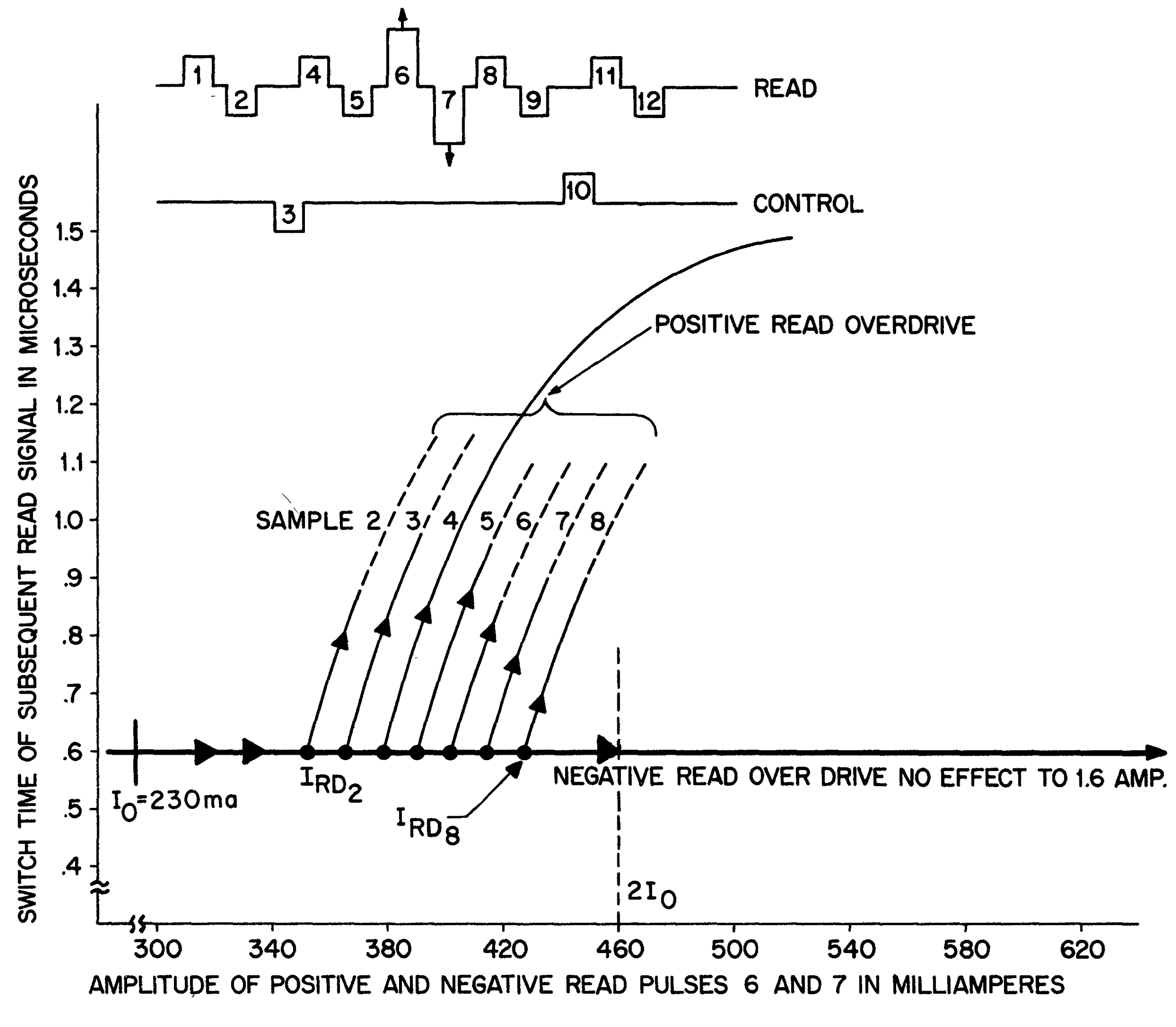


Figure 17. PLOT OF SWITCH TIME FOR "ZERO" READOUT SIGNAL
AS A FUNCTION OF READ PULSE 6 AND 7

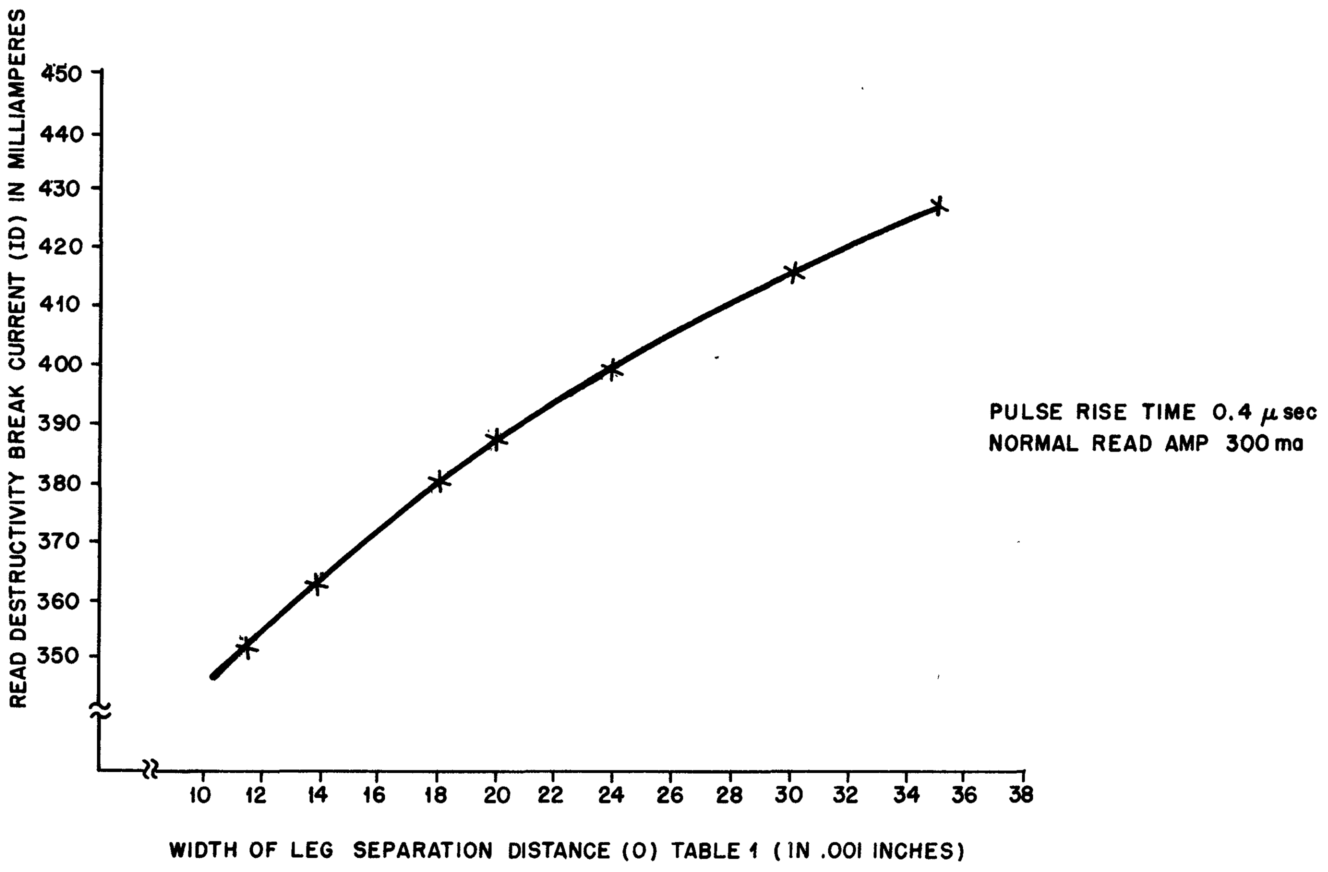


Figure 18. PLOT OF READ DESTRUCTIVITY THRESHOLD AS A FUNCTION OF THE LEG WIDTH SEPARATING READ AND CONTROL APERTURES (NO INNER WALL BIASING TECHNIQUES EMPLOYED)

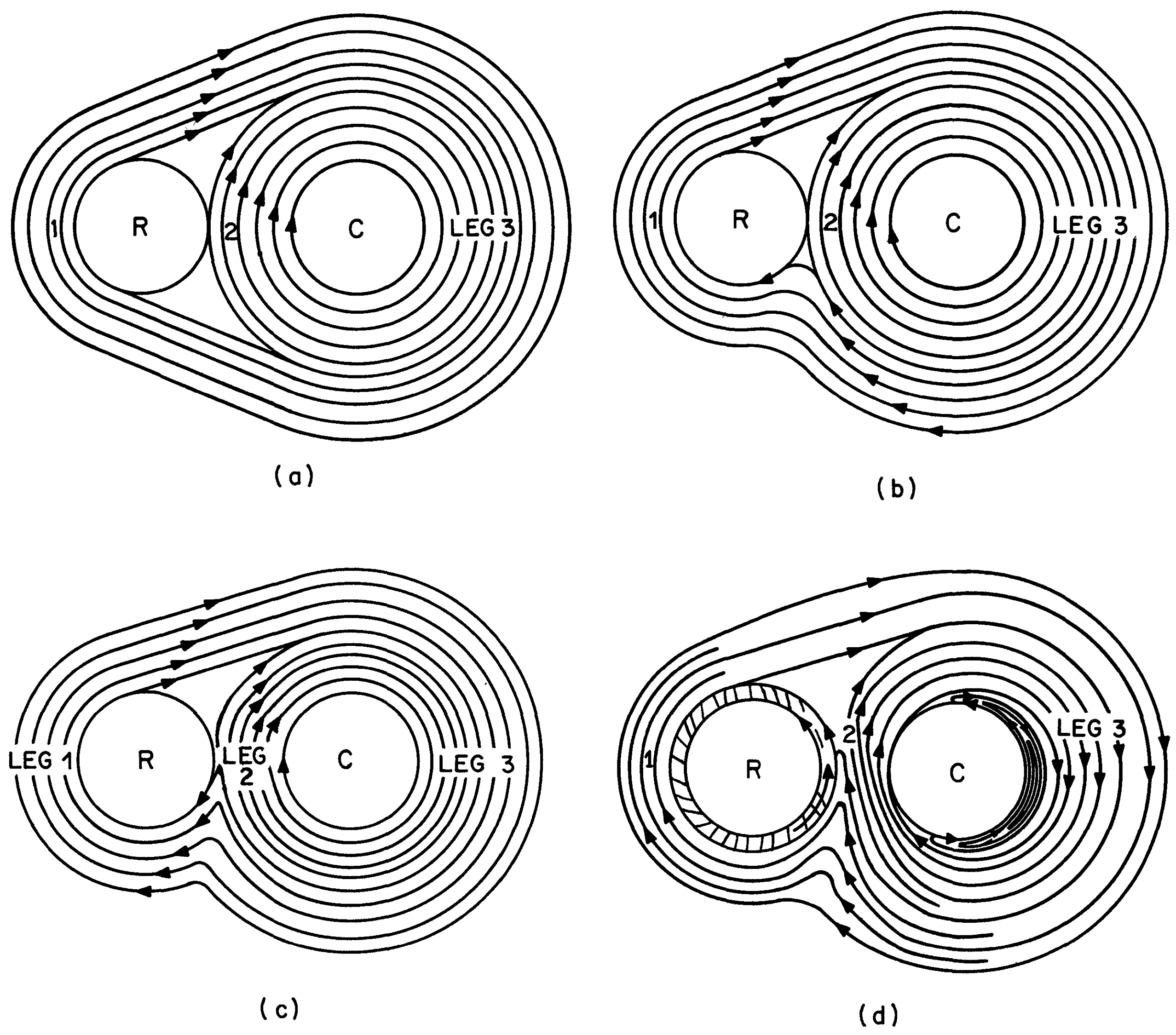


Figure 19. SWITCHING MECHANISM ASSOCIATED WITH READ DESTRUCTIVITY THRESHOLD
(a) No Positive Read Pulses Applied to Read Hole
(b) Positive Pulse Applied to Read Hole Below Destructivity Threshold
(c) Positive Pulse Applied to Read Hole Below Destructivity Threshold (More than 20b)
(d) Positive Pulse Applied to Read Hole in Excess of Destructivity Threshold

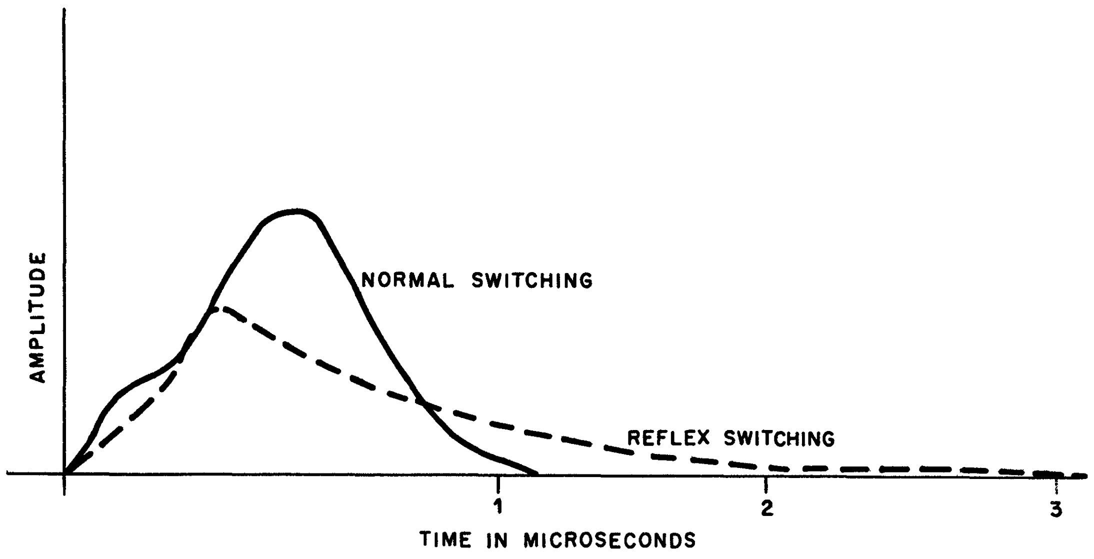


Figure 20. OBSERVED REFLEX SIGNAL COMPARED TO NORMAL SIGNAL

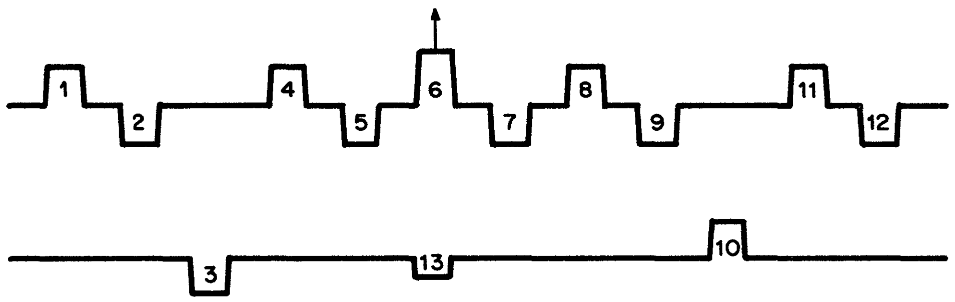


Figure 21. PULSE PROGRAM USED IN PROVING THE REFLEX SWITCHING PHENOMENON

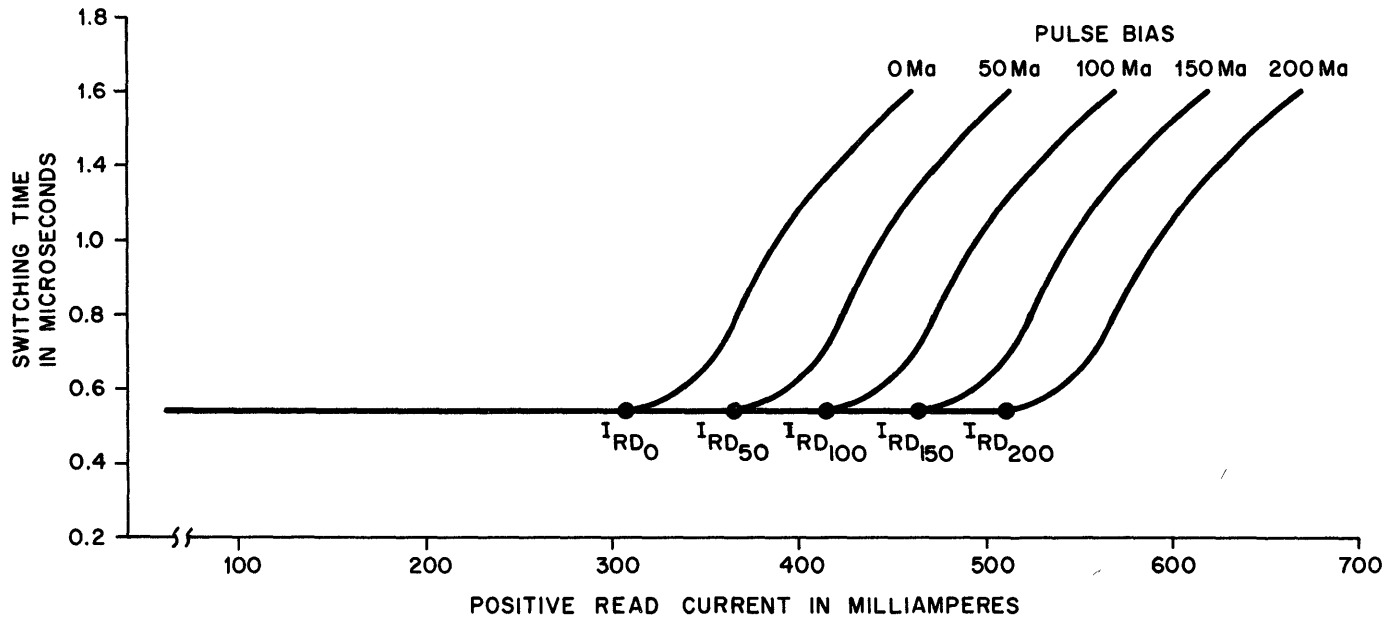


Figure 22. PLOT OF READ DESTRUCTIVITY WITH A PULSE BIAS CONTROL

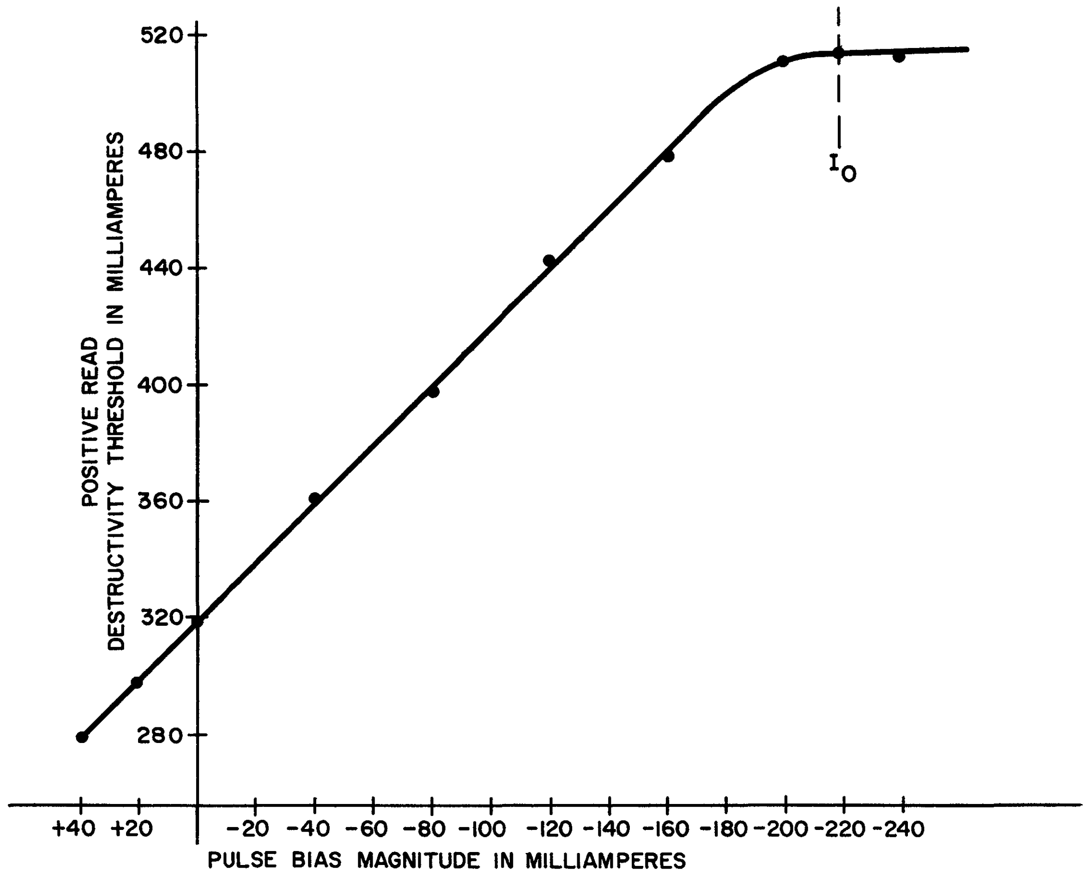
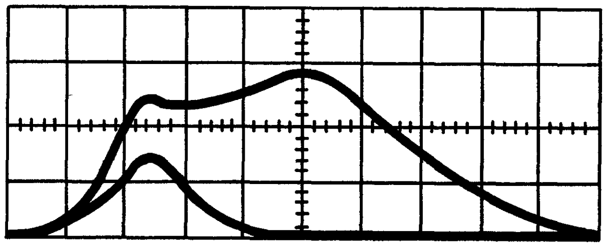
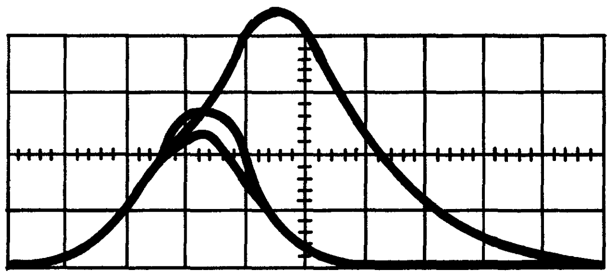


Figure 23. READ DESTRUCTIVITY THRESHOLD VS INNER WALL BIAS


0.2 μSEC / CM
.01 V/CM
(a)


0.2 μ SEC / CM
.02 V/CM
(b)

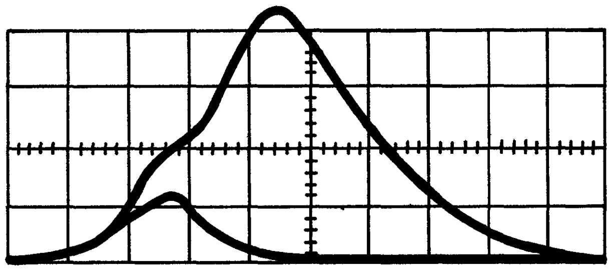

0.2 μ SEC / CM
.02 V/CM
(c)

Figure 24.
EXACT ENLARGED COPIES OF ACTUAL OSCILLOSCOPE TRACES FOR TEST SAMPLE NO. 3

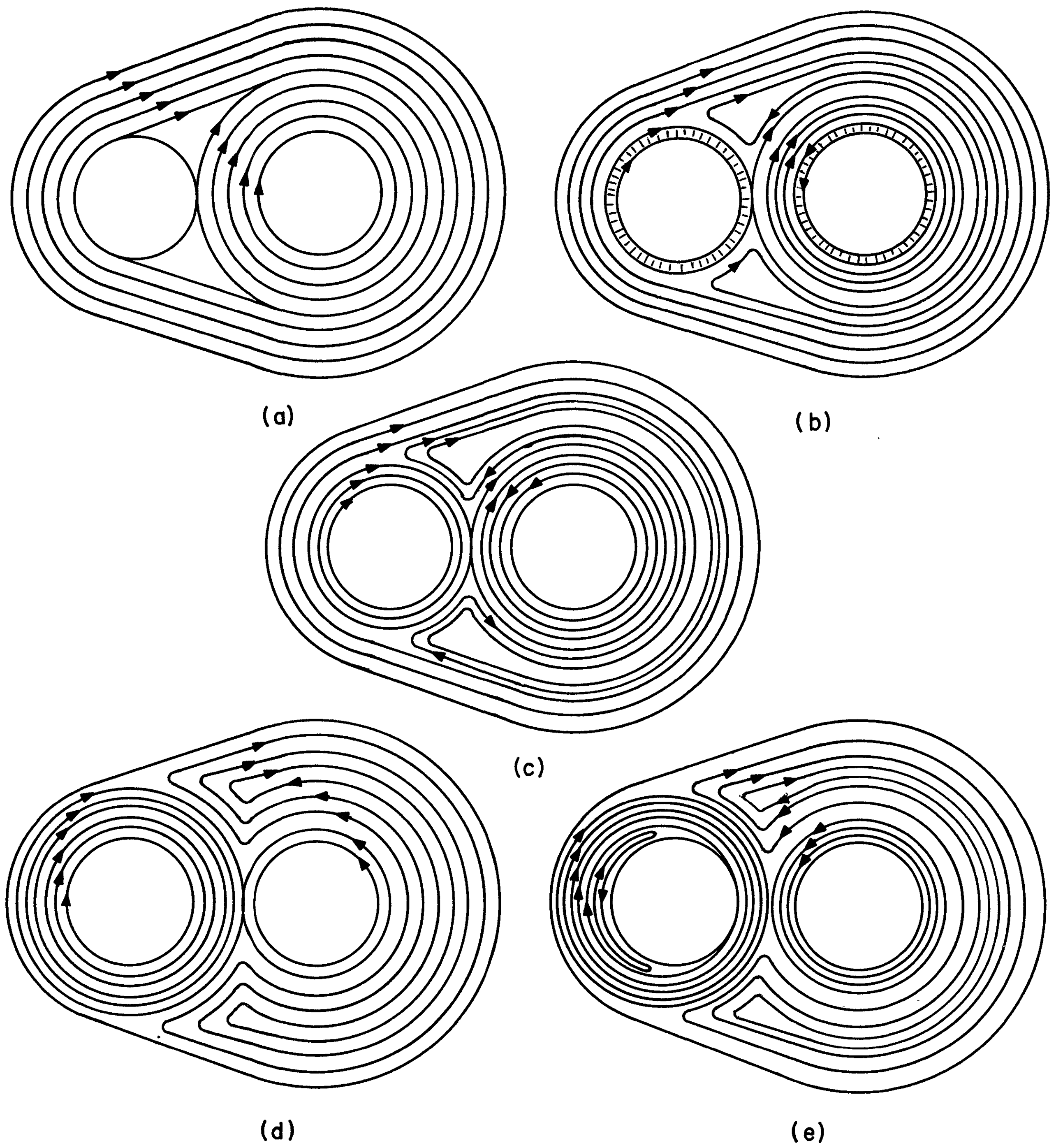


Figure 25. SWITCHING MECHANISM ASSOCIATED WITH WRITE "ONE" CONTROL PROCESS
(a) Write "One" Control Below I_{RDS} Break Point
(b) Slightly in Excess of I_{RDS}
(c) Nearly I_{RDS}
(d) Full written "one"
(e) Slightly in Excess of I_{RB}

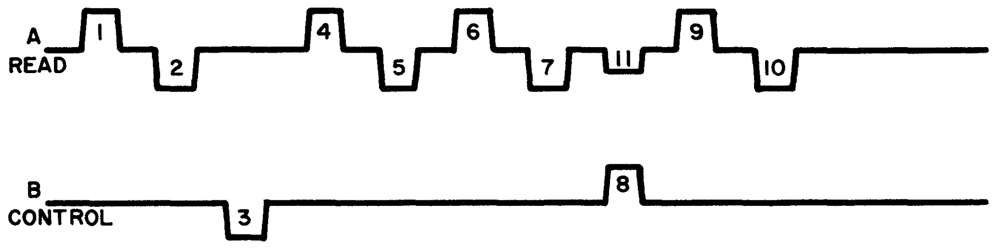


Figure 26. PULSE PROGRAM USED TO BIAS INNER WALL OF READ APERTURE
DURING WRITE "ONE" CONTROL PROCESS

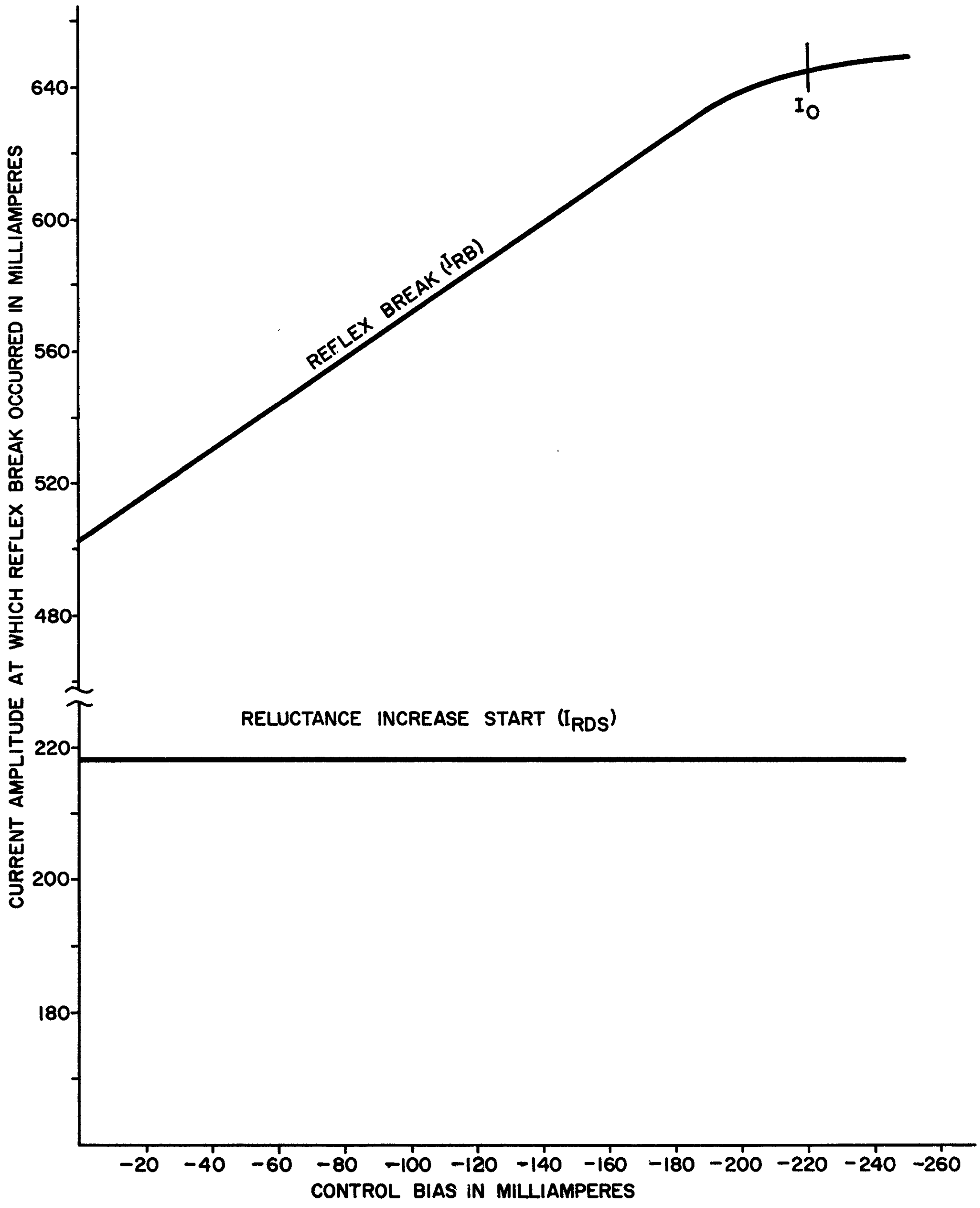


Figure 27. PLOT OF BREAK POINT WITH INNER WALL BIAS

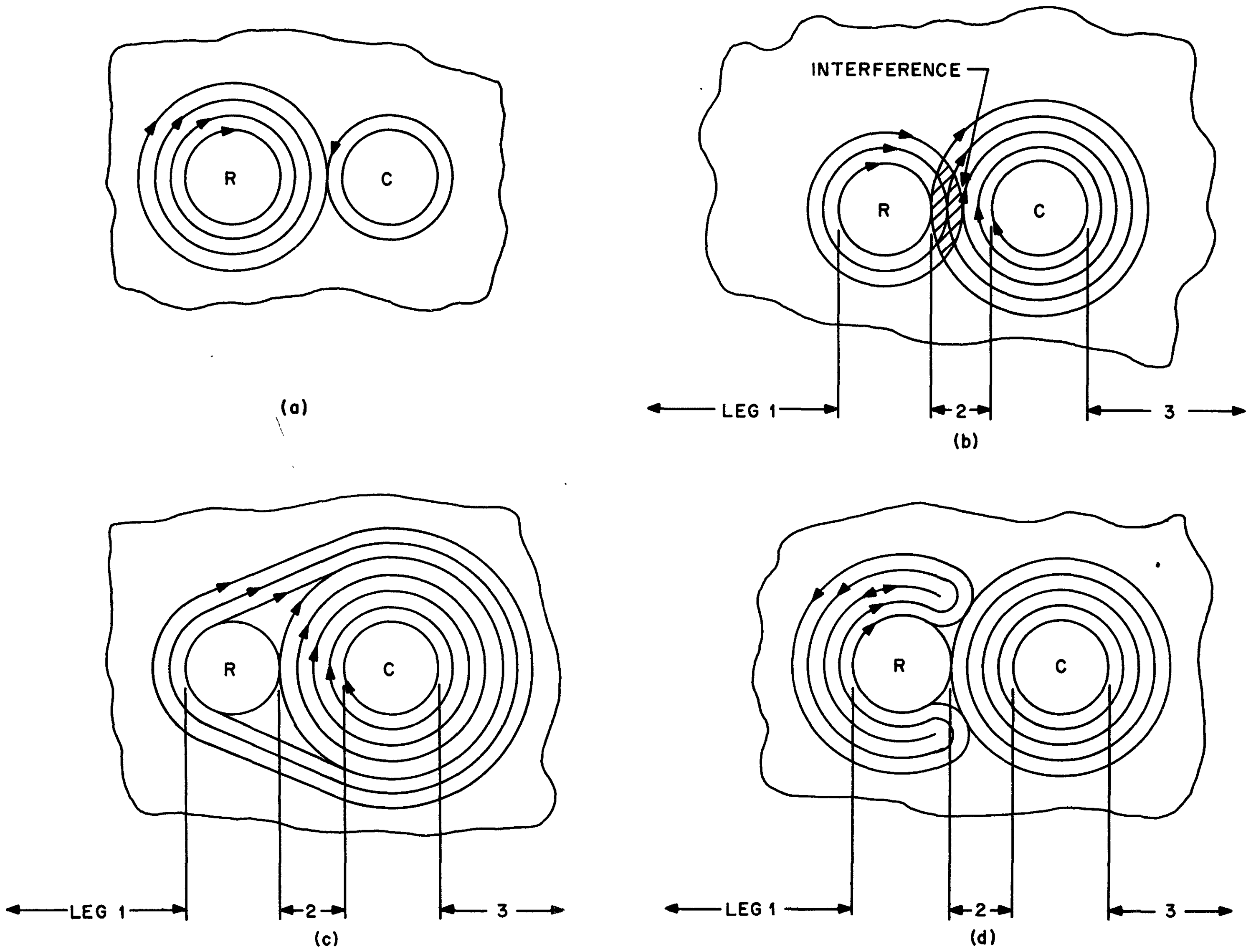


Figure 28. EXPERIMENTAL VERIFICATION OF "PULLEY" FLUX PATTERN

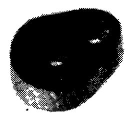


Figure 29.
"TEAR-DROP" MARS DEVICE USED TO VERIFY "PULLEY" FLUX DISTRIBUTION (PHOTO)

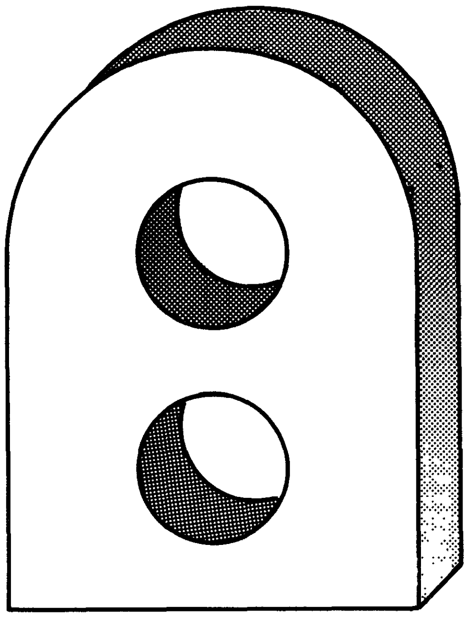


Figure 30. SCHEMATIC OF PRODUCTION "KEY-HOLE" MARS DEVICE

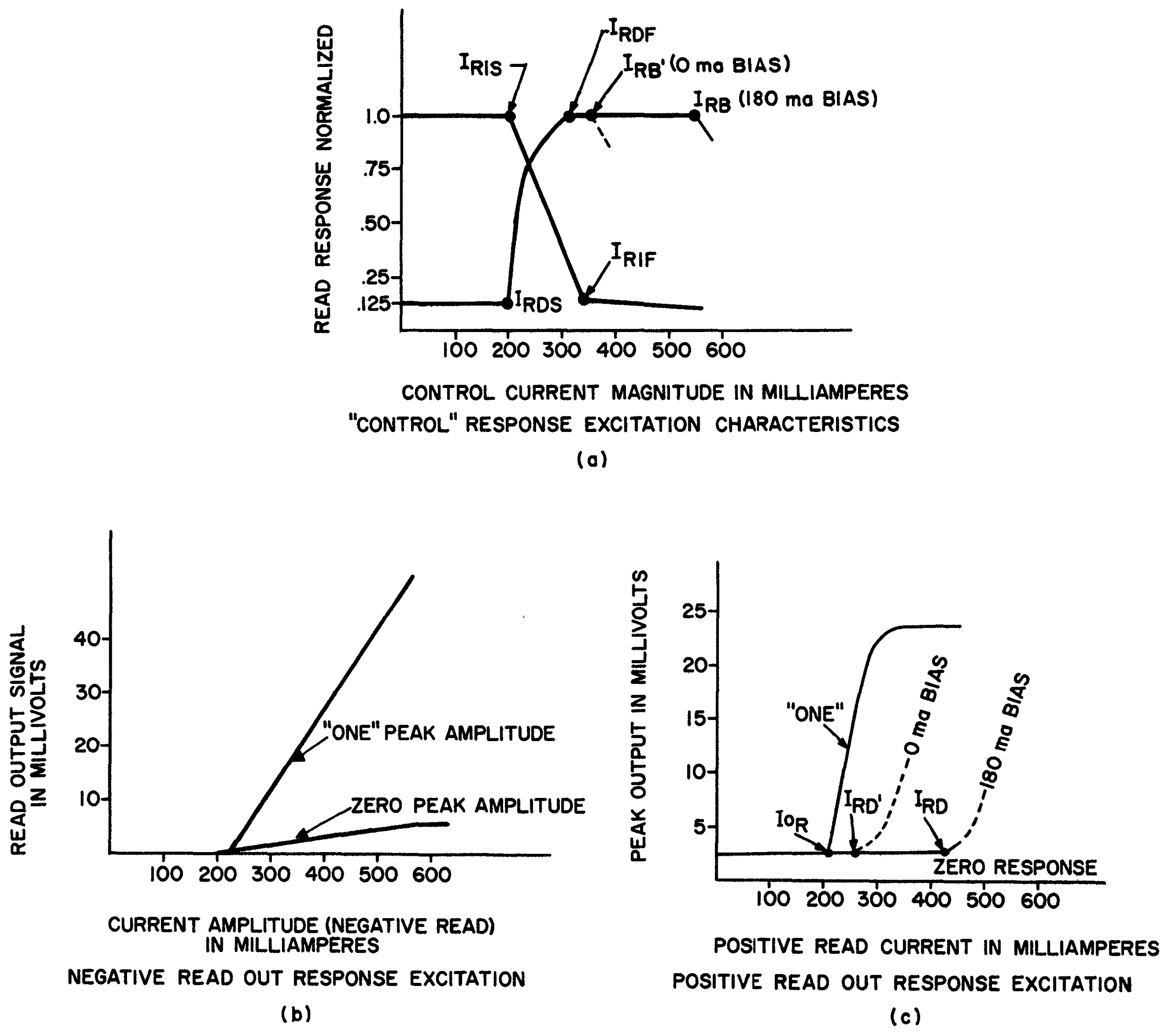


Figure 31.

RESPONSE EXCITATION CHARACTERISTICS OF "KEY-HOLE" MARS DEVICE

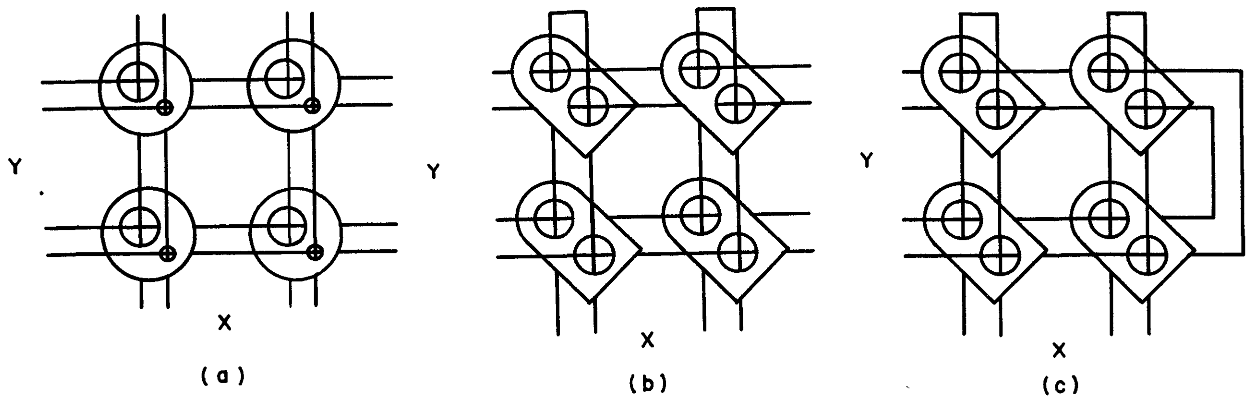


Figure 32. COINCIDENT CURRENT SELECTION SCHEMES
(a) Classic Three-Wire System
(b) MARS Five-Wire System (Sense and Inhibit Windings Not Depicted)
(c) MARS Effective Four-Wire System (Sense and Inhibit Windings not Depicted)

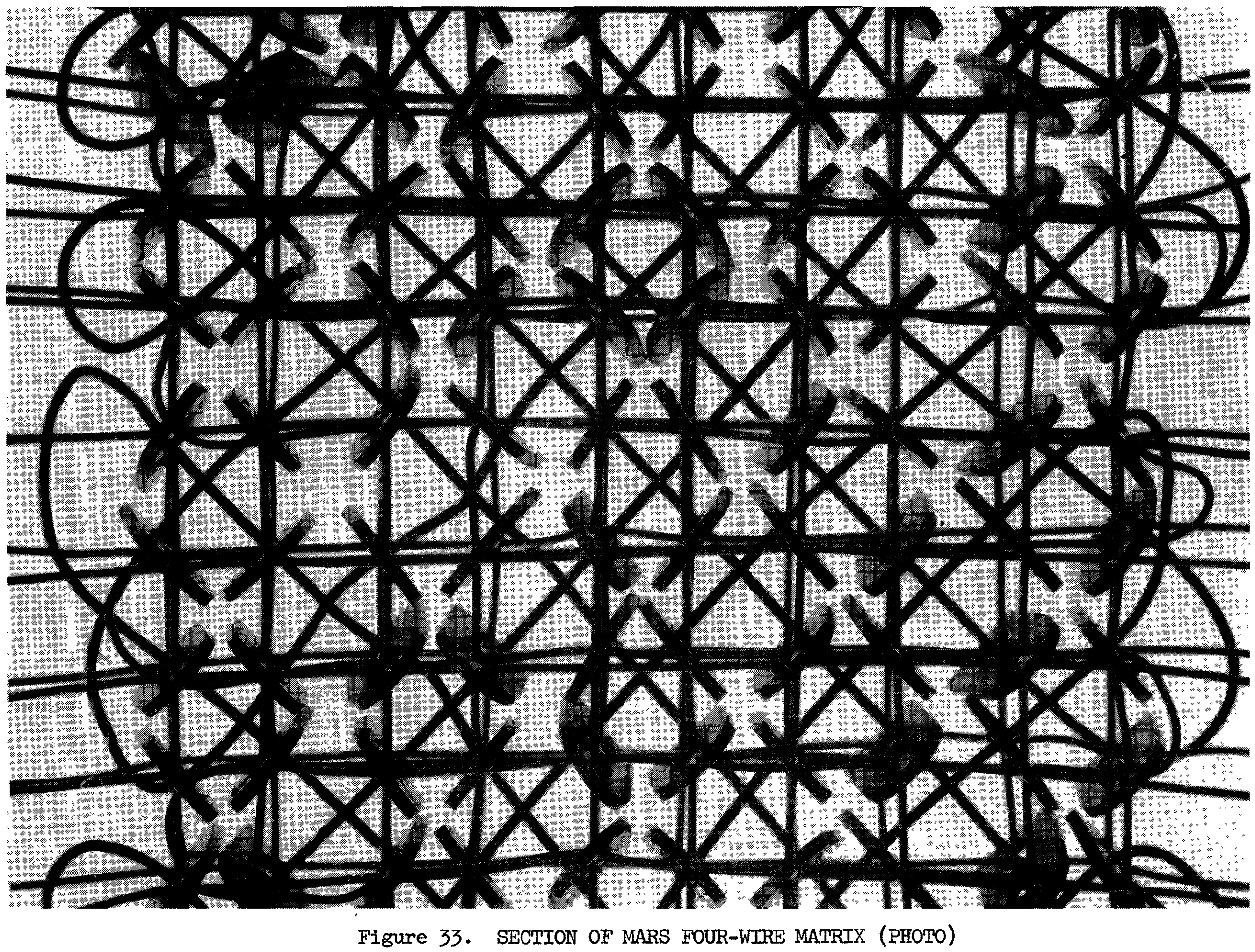


Figure 33. SECTION OF MARS FOUR-WIRE MATRIX (PHOTO)

HIGH SPEED OPTICAL COMPUTERS AND QUANTUM TRANSITION MEMORY DEVICES

Lewis C. Clapp
Computer Development Laboratory
Sylvania, Needham, Massachusetts

I. The Problem

Today we are pressing forward to produce a new generation of computing devices which are to be larger, faster, and more reliable than any class of preceding machines. The avenues which are being taken to follow this approach are as numerous as topics in a modern physics text book such as Parametric Oscillation, Quantum Mechanical Tunnelling, Superconductivity or Thin Film Magnetic Properties. Sophisticated though these approaches may be, they do not come to grips with one simple problem which is becoming more serious with each development in the components field. As we send electro-magnetic energy down a wire there is a tendency for that wire to act as a radiating antenna. This effect also means, of course, that neighboring wires will act as receptors and absorb some of the spurious energy as undesired "noise." While we were operating our computers in the microsecond region one could design around this noise problem using relatively simple and straight forward pulse techniques, but now that we are entering the new generation of ultra-fast computers much more elaborate considerations are necessary. It is not uncommon today to hear respectable computer men talking of wave guides and other microwave plumbing! Meanwhile, the paremetron computer designers speak of placing each unit in multiples of half wave length distances from its neighbors. The steps which must be taken to get around the interference problem are certainly in the direction of less design flexibility and while we have been miniturizing the components the connecting links between them seem to get bigger all the time.

It is one purpose of this paper to ask if there are not other, possibly more pleasant, solutions to the problem and to investigate the class of devices dictated by this approach. Since space is short, we must concentrate only on one aspect of these considerations, namely memory devices; but we shall also broadly outline the features of a different class of computers which comes from these thoughts. It should be clear that we are speaking of devices which are sometime in the future and for this reason the emphasis in this paper is placed on theoretical rather than experimental topics.

II. Optical Computers

The question before us is this: "Is there a method of transmitting information from one point to another at a rate much greater than 10^6 pulses per second, without generating interference on neighboring information channels?"

A little reflection on this question leads us to consider the possibility of using light waves for the propogation of our signals, for it is clear that light pulses can be transmitted free from generated noise by simple isolation techniques. Let us therefore examine a little more closely this intriguing idea of transmitting our data in the form of pulses of light. (Light in this paper is used to include all optical manifestations from the ultraviolet to the infra-red, however, many of the following statements also apply to the microwave region.)

Any digital computer which we build today must possess at least the following three properties or components:

a). A storage element to maintain the data-in a conventional machine - this corresponds to ferrite cores, flip flops and delays.

b). A means of transmitting each bit of information between internal points in the computer - these correspond to wires in conventional machines and waveguides in the newly proposed devices.

c). A decision making element-diode gates, core logic and so on in present day machines.

Storage Device

What we might use as the storage element in such a machine is actually the subject of this paper and will be treated in fuller detail shortly. Here we will simply point out that the type of memory element theoretically discussed in the following paragraphs makes use of well known quantum mechanical properties of radiation. The idea essentially is to use the ground state and certain excited energy levels of our system, for example an atom, as storage positions with the switching action between logical conditions being optically induced transitions from one energy level to the next. We will see that one major advantage of such a method is the inherent fast switching times which can be achieved. The transition time between optical levels is frequently in the order of 10^{-9} seconds and even faster switching times are found if one uses electron spin resonance techniques.

Transmission Lines

To transmit our light pulses from point to point, we can probably take advantage of the fact that certain materials will transmit light along preferred axes for relatively long distances with very little attenuation. Perhaps the most interesting of these media are long drawn out glass fibres which are extremely flexible and easy to use. Although there is some attenuation in passing a light beam down such a fibre practically all this loss is due to absorption by the glass. This absorbtion amounts to less than a quarter of one percent of energy per inch. Generally the optical fibre can be coated by a thin glass film of lower refractive index to eliminate "cross talk" between neighboring transmission lines. The low cost of these glass fibres compares favorably against the expensive wave guides needed for micro-wave computers.

The science of fibre optics is now a rapidly developing subject and is well treated in the literature.[1, 2]

Decision Elements

By admitting the possibility of computing with radiation as the carrier of information, the door is open to a whole new class of decision making techniques. There are countless properties of light beams alone or light in interaction with crystals and other matter that could be used to generate logic. To consider a very simple example, let us recall that light can be polarized in several ways; linearly, elliptically and circularly. In fact, a beam of circularly polarized light can be either right circular polarized or left circularly polarized - that is to say, the electric vector of the light beam rotates to the right or left in the plane perpendicular to the axis of propogation. Next imagine all the ways two beams of circularly polarized light can be combined. The results are summarized in table one and are strongly suggestive of a logical and truth table. One will observe that the analogy is not exact for when left and right are combined the result is a linearly polarized beam. This difference is not serious since linear polarized light will not effect future logical operations. Another way to look at this difference is that with three types of polarized light beams - linear, circular right, and circular left; we can generate ternary logic rather than the binary logic in vogue today. If we try to combine more than two beams simultaneously, we find the result leads to a majority decision logic much the same as parametron logic discussed elsewhere in the literature[3].

Since all the input equipment for computers is electro-mechanical, we should also consider gating techniques that will permit us to convert from an electrical impulse to an optical signal. One such method employs the Kerr effect, which states the polarization vector of light passing through certain crystals can be rotated through the application of an electric field. The magnitude of the rotation depends on the crystal material, the length of the light path and the strength of the electric field. Ordinary polaroid filters could be used to monitor the outputs of such a conversion device.

Of course, the converters needed to translate our optical signals to electrical pulses for stimulating the output equipment tied to the computer could be just a simple photocell.

The photocell is a slow component but certainly it is fast enough to keep up with the present output devices available. In addition, considerable work is being carried out these days toward devices which will rapidly convert optical signals to electrical pulses and the results of such research could be directly applicable in this area.

III. Quantum Level Memory Devices

Gaseous Model

It has already been pointed out that the memory devices proposed in this paper center around the basic fact that electrons can make transitions from one quantum mechanical level to another in very short times on the order of 10^{-8} or 10^{-9} seconds at optical frequencies. Now it is well known from atomic theory that the electrons surrounding an atomic nucleus do not have a continuous band of energies but in fact can exist only at certain discrete energy levels. The energy of an electron in the n th level is given approximately by:

$$E_n = \frac{me^4}{2h^2n^2} \qquad (1)$$

where m and e are the mass and charge of the electron respectively and h is Plank's constant. The state of lowest energy corresponding to the principle quantum number n equal to one, is commonly called the ground state. In the absence of any external stimulation most of the atoms will be in the ground state, however, a small portion of the population will be at higher levels at any given moment. The number of atoms in the E_i th energy level is given by the Boltzmann distribution law

$$N_i = Ce^{-\frac{E_i}{kT}} \qquad (2)$$

where k is the Boltzmann constant, T is the absolute temperature of the sample and C is a constant. The normal distribution is plotted in figure one.

For each energy level there are actually n sub-levels which are denoted by a second quantum number ℓ which can assume the integer values from 0 to n-1. Each sub level may not have the exact energy given by (λ) since there is a slight energy dependence on 1 as well. In figure two we show an energy level diagram for Hydrogen gas. In spectroscopic notation one generally denotes the values of ℓ equal to 0,1,2,3,4 etc. by s,p,d,f respectively. Thus, the 1s level refers to the ground state where n is 1 and ℓ is 0. Some transitions are indicated by the solid diagonal lines in the figure.

The hydrogen atom can absorb incident radiation which will cause the electron to go to a higher energy state, and it is the frequency of this incoming radiation which will determine the resultant energy level which the electron achieves. In general, a transition from level E_m to E_n requires a radiation frequency

$$\nu_{nm} = \frac{E_n - E_m}{h} \quad (3)$$

Conversely, an electron in a higher state will spontaneously radiate down to a lower enerby level through the emission of radiation whose frequency is again given by (3). Referring again to figure two, one will observe that transition between all sub-levels do not occur. Some of the "forbidden" transitions are indicated by the dotted lines in the figure and fail to take place because of a quantum selection rule which only permits transitions in which ℓ changes by $\pm$ 1. From these facts one can readily conclude that if an electron falls into the 2s state it is "trapped" in the absence of external radiation. The only state of lower energy is the 1s state and this cannot be reached since ℓ would not change by unity in such a transition. Such states are frequently called metastable. For a more detailed review of atomic spectra theory the reader can refer to Herzberg[4] or Richtmeyer and Kennard[5].

Consider a simple version of a device which can be used to store data through the principles outlined above (fig. 3.). A small vessel with two electrodes encapsulates a gas with known metastable states and a second gas with an ionization potential well below the metastable state energy. The second gas merely provides free electrons which will impart energy to metastable gas atoms during the reading and writing processes. When our gas atoms are in the normal energy distribution, mainly the ground state, we shall say that a logical zero is stored in the cell. A voltage pulse V, will cause the free electrons to excite the gas atoms, to higher energy levels after which many will radiate down to the 2S states. The remainder will return primarily to the ground level. Those electrons in the 2S metastable level will remain trapped until a small read pulse v is applied. This will result in perturbation of the 2S electrons through collision with a consequent induced transition back to ground and a corresponding emission of radiation. This radiation could be detected by an external device. If the memory cell were in the zero state, our interrogation pulse would only shift the energy level of the ground state electrons slightly and no significant output radiation would result. It is clear that the same actions can be caused by admitting radiation into the cell to effect the transitions from ground to the upper levels and to induce transitions of trapped electrons to ground. The radiation method is actually simpler and is compatible with our concept of optical computers.

Utilizing this technique, one could construct a large scale memory in the following manner. Two glass plates are joined in a gaseous environment so that the upper plate which contains many spherical half cavities encloses the gas (fig. 4). In the lower flat plate may be three electrodes which will be used to excite the gas in each cavity during read and write operations. Each cavity can be activated at the appropriate time through the mechanism described in the preceding paragraph. The radiation output which occurs on the detection of a stored logical "one" could be channeled through an optical fibre to a photocell associated with a particular group of bits. The electrodes in each storage cell have been included here strictly for simplicity of description. One could also excite the atoms in a cell through radiation thus eliminating the need for electrodes and all attendant mechanical construction problems.

So far we have neglected to point out a number of deficiencies in the gaseous metastable memory model just described. Perhaps the most serious difficulty is that metastable states in gases are not quite as permanent as we may have implied. Sometimes even forbidden transitions will occur very weakly but never-the-less, strong enough to depopulate the trapped upper energy levels after a long time interval. Collisions between the atoms which constitute the gas will also induce spurious transitions to the ground state from the upper levels. To reduce the effect of atomic collisions one could work with lower pressure gas systems at reduced temperatures but such an attempt would also result in lower amplitude output signals. Finally, the technological and production problems associated with gaseous devices are undersirable in their own right even if the other difficulties can be minimized or eliminated. For this reason we shall next turn to the task of looking for similar fast switching storage action in solid state materials. Before taking up this question, one final remark is in order. It is tempting to consider the possibility of artifically prohibiting transitions between certain energy levels through the application of external electric and magnetic fields. Although this idea has not been fully investigated as yet, certain details are already well known from basic quantum theory. First, application of a pure electric field has the net effect of simply displacing all the energy levels (stark effect), wheras a pure magnetic field will generate third order sub-levels for each level shown in figure 2, corresponding to the third or magnetic quantum number associated with the particular atomic system (Zeeman effect). Neither an electric or magnetic field alone would have the desired effect, but some combination of the two might work for a given gas.

A Solid State Model

We now turn our attention to similar phenomena in solid state materials in an attempt to overcome the problems encountered in the gaseous version of the proposed memory device. One reason for considering solid state materials is that molecular collisions can often be reduced, but this, of course, requires cooling of the material and may in itself be objectionable. In the domain of solid state physics we are fortunate in finding scores of effects with potential usefulness as computer memory devices. It will be recognized however that most of the effects in current solid state technology make use of charge migration; that is the physical motion of electrons and holes from one part of the crystal to another. Such motion is too slow for the speeds which we are considering and therefore the emphasis will be placed on motion independent phenonema.

F Center Trapping

The F center is only one and the simplest of several types of color centers now known to exist in various ionic crystals. It is generally thought to be a lattice point with a missing negative ion. The resulting vacancy then acts as a positive charge center with the capability of trapping electrons. A trapped electron is therefore held in place in a manner quite analagous to the way electrons assume orbits around atomic nuclei. Of importance to us is the fact that such electrons are loosely bound and can be easily freed either through the action of external radiation such as infra-red light or by simply heating the crystal. The relation of F centers to the crystal band structure is depicted in figure 5. For proper operation as a memory device the crystal properties must be such that the Fermi level at the operation temperature is below the F center energies.

The dynamics of a memory element using F center action is relatively uncomplicated. In the normal condition, the electrons will be below the Fermi level and therefore below the F center traps as well. Such would be the "zero" state of the memory cell. Writing a "one" into the unit cell is accomplished through illumination of the crystal which causes the electrons to rise into the conduction band. If the F center density is high enough, a significant number of electrons will be trapped, the remainder of electrons falling back into valence band. Detection of the particular state of a memory cell can be handled in several ways. First, we may free the trapped electrons by irradiating the crystal with short wave-length radiation or possibly by applying an electric field; in either case attempting to observe the transition radiation as the electrons return to the valence band. If the crystal parameters are exactly favorable such a method will be acceptable. On the other hand, a long search may be required to find a crystal with such a detailed structure. One may even be able to produce the proper crystal taking advantage of the recent work in F center production in crystals through X ray irradiation. See for example the current paper of Mitchell[6] and the general statements in Kittel[7].

An alternate method of continuously monitoring the state of the crystal would take advantage of the fact that F centers are themselves color absorbers. A beam of weak radiation passing through the crystal in the zero state will be absorbed but a corresponding beam of radiation passing through a crystal in the one state will find the material transparent and go out the other side. This monitoring technique is attractive since it does not necessarily destroy the stored information during the reading process. A large scale memory of this type could be made by vacuum coating the F-center abundant material onto a substrate to form a large wafer. Etching is then used to isolate the individual memory cells into a grid-like arrangement for independent memory action (fig. 6). This wafer might even be placed over an electro-luminescent material to increase the number of lumination generated electrons capable of being captured by the F centers. This type of construction would result in inexpensive memory wafers with high bit densities.

We come now to the problem of selecting a particular bit on our wafer. If we wish to use only optical signals the following technique seems satisfactory. Input lines, which are optical fibers, are applied as X and Y drivers to an electro-luminescent switching matrix (fig. 6). The properties of this matrix are such that an optical signal must exist both at drivers X_1 and Y_1 for an output, which is again an optical signal, to appear on selection line S_{11}. The selection lines are themselves optical fibers which go directly to the chosen memory cell to produce the desired storage or read out action.

One sould not neglect to make a comprehensive study of other types of color centers also, as potential mechanisms behind this type of data stbrage device. For while it is evident that many of our conclusions about F center memories are tentative and await further experimental verification, the possibilities are extremely attractive and deserve further investigation.

Maser Models

We have not as yet pointed out the interesting similarity between the memory devices discussed so far in this paper and certain elements of the maser theory. The maser is well known as a system for producing high signal amplification with very low noise background. First, we shall briefly review the basic principles of two level maser theory.[8,9] Recalling figure one, the Boltzmann distribution tells us how the various energy levels of a material will be populated under normal conditions. It can be seen from the figure that the lower level E_A is more substantially populated than the upper levels, say E_B. In a maser, one attempts to reverse this situation with a pair of levels, in such a manner that the upper state has the greater population. Shortly after this population inversion, another signal is used to force these excited electrons or spin states back down to the E_A level. The resulting emission from the downward transitions becomes the output signal. If resonance factors are properly introduced into the system high orders of signal amplification can be produced. Once the population has been inverted, there is a tendency for the electrons to spontaneously drop back to the normal distribution without external stimulation, the time required for this process to go to completion being referred to as the "relaxation time". Usually, this relaxation period is several orders of magnitude slower than the transition times involved, but the exact values may vary under different operating conditions.

Let us try to interpret the maser model as an information storage device along the lines outlined in the foregoing pages. The basic feature of all the preceding models, transitions between quantum mechanical levels, is equally important for maser operation; but we may expect some difficulty in retention of the data. Consider then a simple two level maser as a one bit storage device. When the distribution of energy states is closely that of the Boltzmann formula (2), we shall say the cell is storing a logical "zero". Assume next that a write signal is admitted with sufficient power to invert or equalize the level distributions (figure seven), and call this the "one" state. If the relaxation times are long we will have a pretty fair memory element. There are again at least two ways to sense or read out of such a memory cell. In the first method when it is desired to read, one can stimulate emission from level B to level A by an external signal. If the cell is in the "one" condition a large output signal will be detected. Conversely, if the memory unit is still in the zero state there would be no downward transition and consequently no output signal. This read out technique has the advantage of falling in line with usual maser practice, but the serious disadvantage of being a destructive read out technique and would require elaborate rewrite procedures to conserve the data. The second possibility may be more appealing. If a weak signal of frequency ν, where

$$\nu = \frac{E_B - E_A}{h} \qquad (4)$$

is sent through the maser, while in the normal distribution, absorption takes place. The material then said to be opaque for radiation of this frequency. The physics behind this absorption is easy to understand since the energy carried by the interrogating signal is exhausted in raising a few electrons from E_A to the higher E_B level. If the same signal is sent through a maser in the inverted population condition, the upper levels will already be filled and no such absorption can occur, that is the crystal will be transparent. Therefore, one can continuously monitor the logical state of the unit without destroying the stored information.

As mentioned above, relaxation effects will play an important part in the operation of such a maser memory. We cannot generally expect the data stored in our cell to persist indefinitely for the population will eventually return back to the Boltzmann distribution. By choosing materials with long relaxation periods one can hope to do a large number of computer operations before the data deteriorates badly, however, a reconstitution procedure must finally take place. There are a number of methods with which one can face this difficulty and the most straightforward techniques will be mentioned here. Two maser cells connected in series and operated in two phase action, could be employed for the storage of each single bit (figure 8). During phase one time data can be written into or read from the first of these memory cells while a normalizing pulse is returning the second cell back to the Boltzmann distribution. The duration of phase one will naturally be shorter than the relaxation time of the crystal. A short interval occurs after phase one which is used to transfer the stored data from the first to the second memory cell. The same operations occur during and after phase two except that now the information is shifted back to the first unit. A crude timing scheme for these operations is given in figure nine. Since the maser is essentially an amplifier such a method of dynamic data preservation can be continued indefinitely. Several other methods of getting around the relaxation problem have been proposed, should the two maser cells per bit scheme prove to be undesirable. First, there is the idea mentioned briefly before, of prohibiting transitions between quantum levels by the application of external conditions such as electro-magnetic fields. The earlier comments apply to the maser case as well. Secondly, in place of the second cell one might substitute some sort of pulse delay unit, so that at the end of the useful operating period an output pulse would be delayed long enough to return the maser back to the normal condition. The pulse would then be fed back to the maser as an input. We have investigated one such pulse delay unit which also makes use of the gaseous metastable action but our conclusions are still tenative and the device will not be discussed here. The third possibility of beating the relaxation problem is perhaps the most interesting. One could use a three or four level maser and store each bit of data in a single cell. We would then work alternately between various pairs of levels, storing the data in one set of levels while the remaining states were being normalized for the next cycle. Although the details of this multi-level maser principle have been worked out in a cursory manner, the pumping scheme is rather involved and the results are still too preliminary for further treatment at this time.

In our treatment of maser memory models so far, we have considered only masers in which the switching occurs between energy levels. It is well known however that many of the new masers are based on electronic spin resonance or optical pumping between the hyperfine levels of a gas or crystal. There are several advantages to be gained by using optical pumping between these hyperfine levels in a memory system. First, the transition times involved are generally an order of magnitude faster. Secondly, in a pumping scheme such as those discussed by Kastler[10,11], the appropriate transitions are only produced if the incoming radiation is properly polarized. That is to say, we could construct a memory in which the storage action would occur only if the incident beam had the right type and degree of polarization. This reminds us again of the earlier remarks on optical gating with polarized beams and we see now that this type of decision technique can blend nicely with an optical maser memory.

Now that the possibility of using masers as a storage element has been outlined, a few words of caution are in order. Masers, as they exist today, are large, bulky devices often requiring considerable external equipment such as cooling systems and magnets. The trend in maser research has been for the production of higher amplification factors. In contrast, the computer designer wants more compact systems with fewer external attachments. Most important, we are looking for maser cavities with a large number of independent cells for data storage. In other words, we are willing to sacrifice amplification for high density storage at lower costs. It can be expected that maser research with these new goals in mind will eventually lead to positive results.

Acknowledgements

Among the many people to whom I owe gratitude for this work, I wish to explicitly mention Dr. H. Yilmaz, H. Cohen and W. S. Lee for interesting discussions and valuable advice, and W. Congleton for his constructive criticisms of the paper. I belatedly thank Dr. J. Cordua and the entire staff of the Laboratory for Electrical Investigations at the University of Chile, for their hospitality and the privelege of working with them in 1959.

References

1. "Fiber Optics; Parts I, II, III and IV" by N. S. Kapanathy, Journal of Optical Society of America, vol 47, 1957

2. "Fiber Optics" by N. S. Kapanathy; Scientific American, vol 203, #5, Nov. 1960.

3. "High Speed Computers" by J. A. Rachjmann; Proceedings of the Eastern Joint Computer Conference, 1959

4. "Atomic Spectra" by G. Herzberg; Dover Publications, 1944.

5. "Introduction to Modern Physics" by Richtmeyer and Kennard; McGraw-Hill, 1947.

6. "Formation of F Centers in KCl by X rays", Mitchell et al, Physical Review 121, 2 p. 484, Jan 15, 1961.

7. "Introduction to Solid State Physics" by C. Kittell; John Wiley, 1956.

8. "Masers", J. Weber; Reviews of Modern Physics, vol. 31, #3, July 1959.

9. "Elements of Maser Theory", A. Vuylsteke; Van Nostrand, 1960

10. A. Kastler, J. Phys. Rad. 11,255 (1950)

11. "Optical Pumping and Related Effects", J. Brossel; appearing in "Quantum Electronics" edited by C. Townes, Columbia Univ. Press, 1960

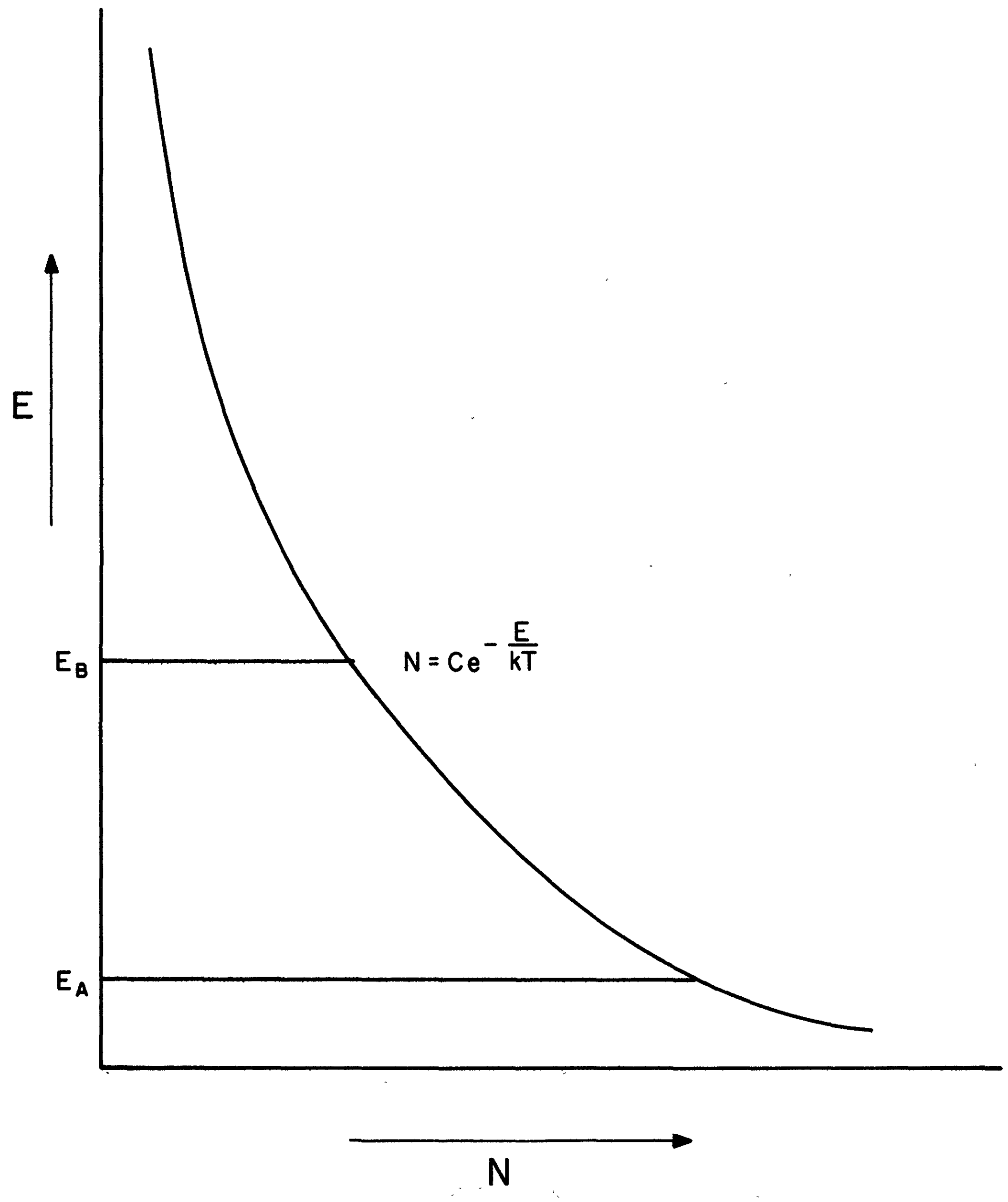


Figure 1. NORMAL BOLTZMANN DISTRIBUTION

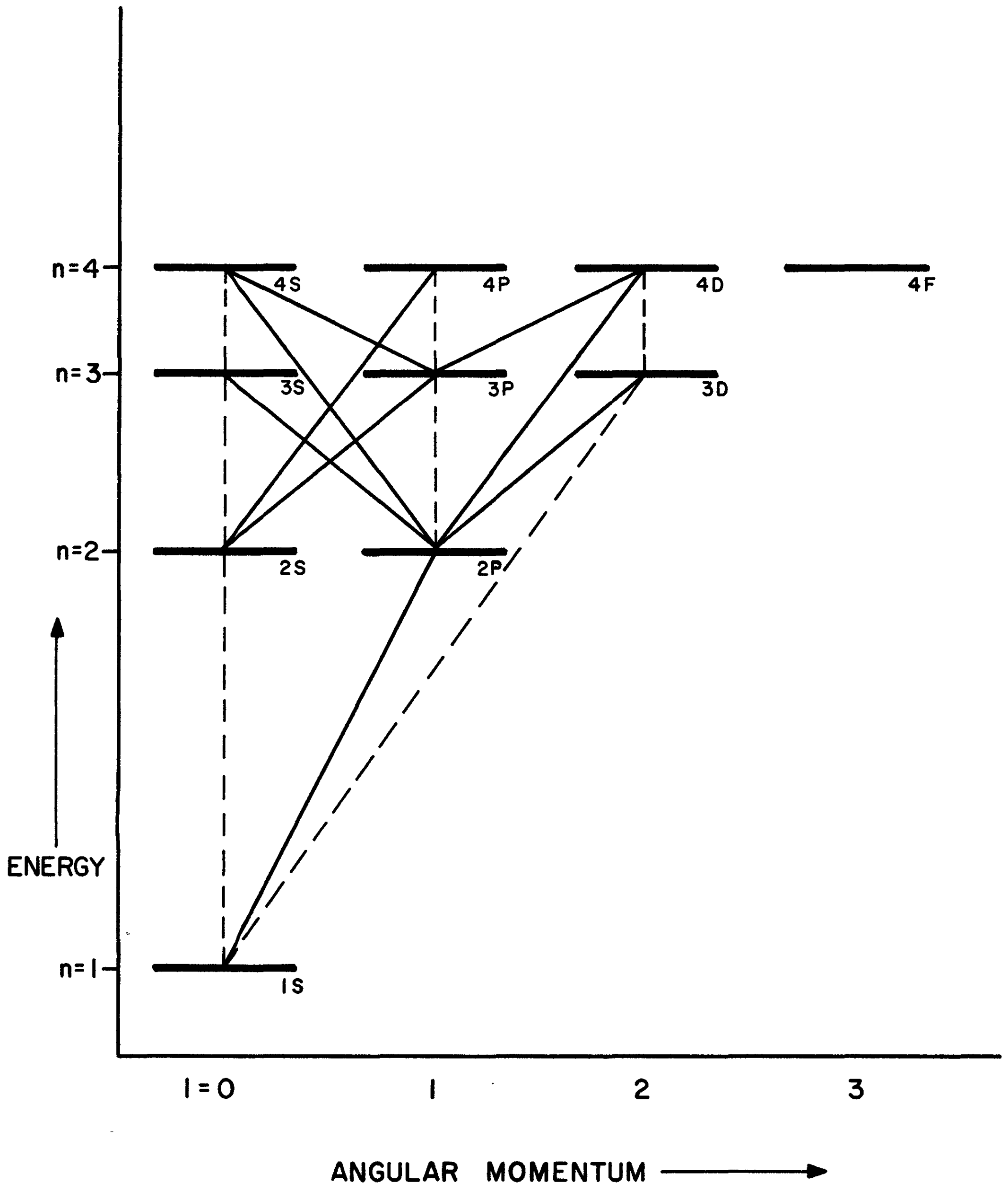


Figure 2. ENERGY LEVEL DIAGRAM FOR HYDROGEN

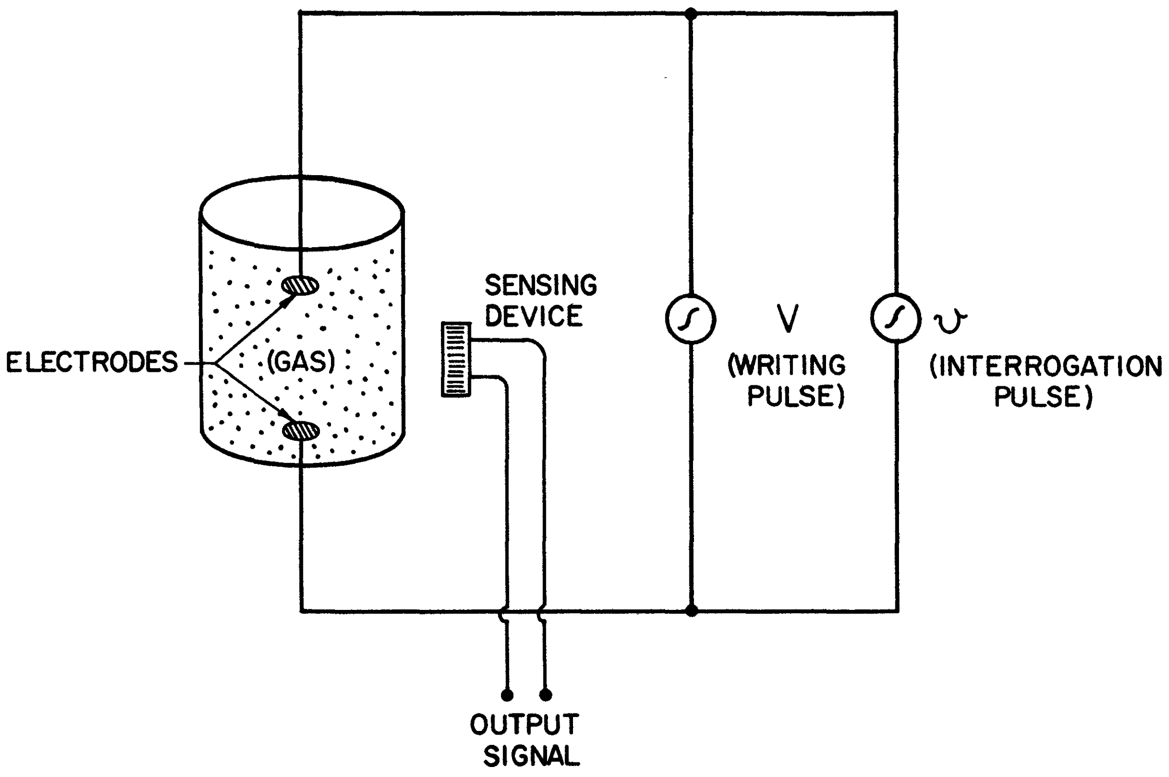


Figure 3. SIMPLIFIED METASTABLE MEMORY ELEMENT

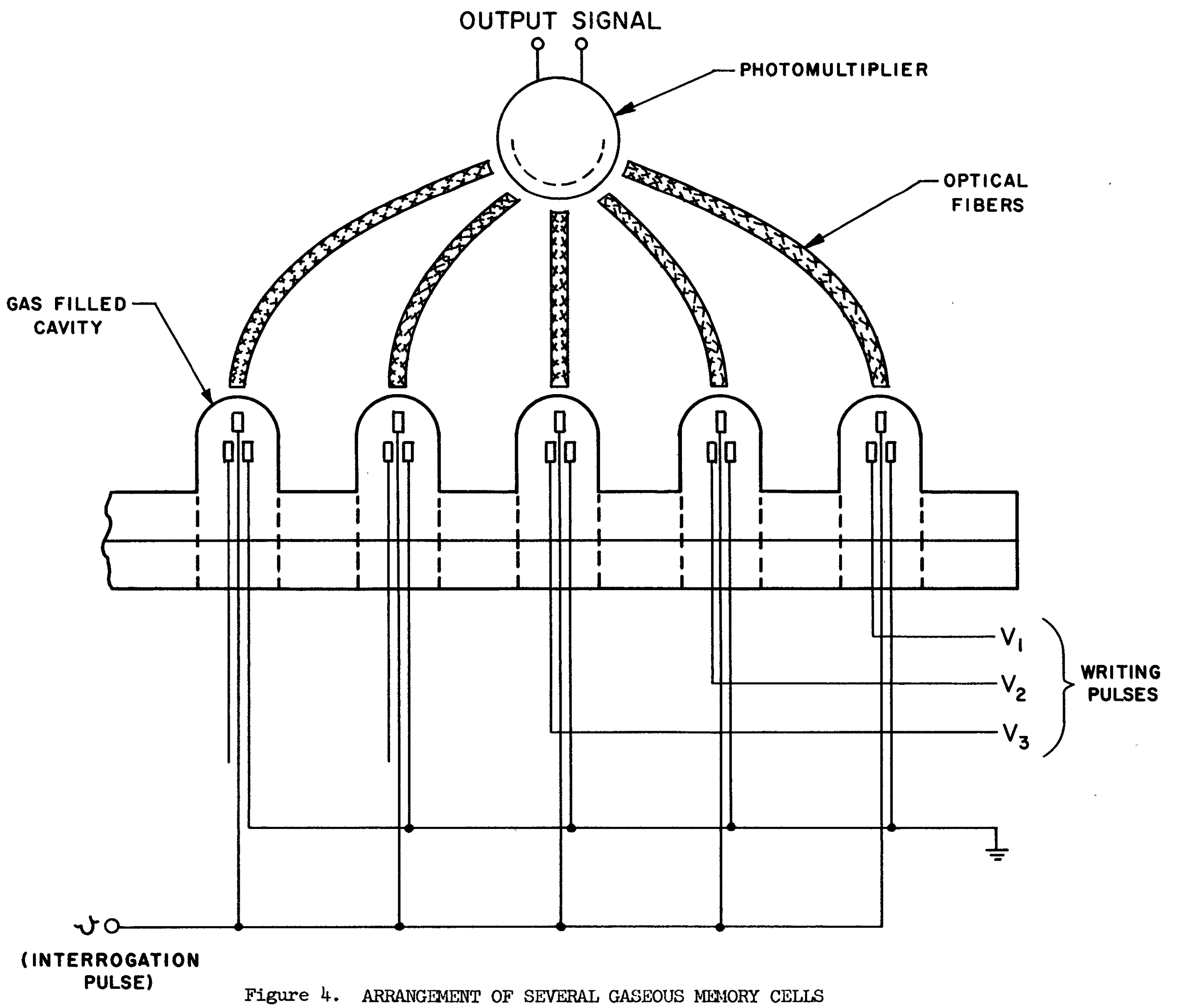


Figure 4. ARRANGEMENT OF SEVERAL GASEOUS MEMORY CELLS

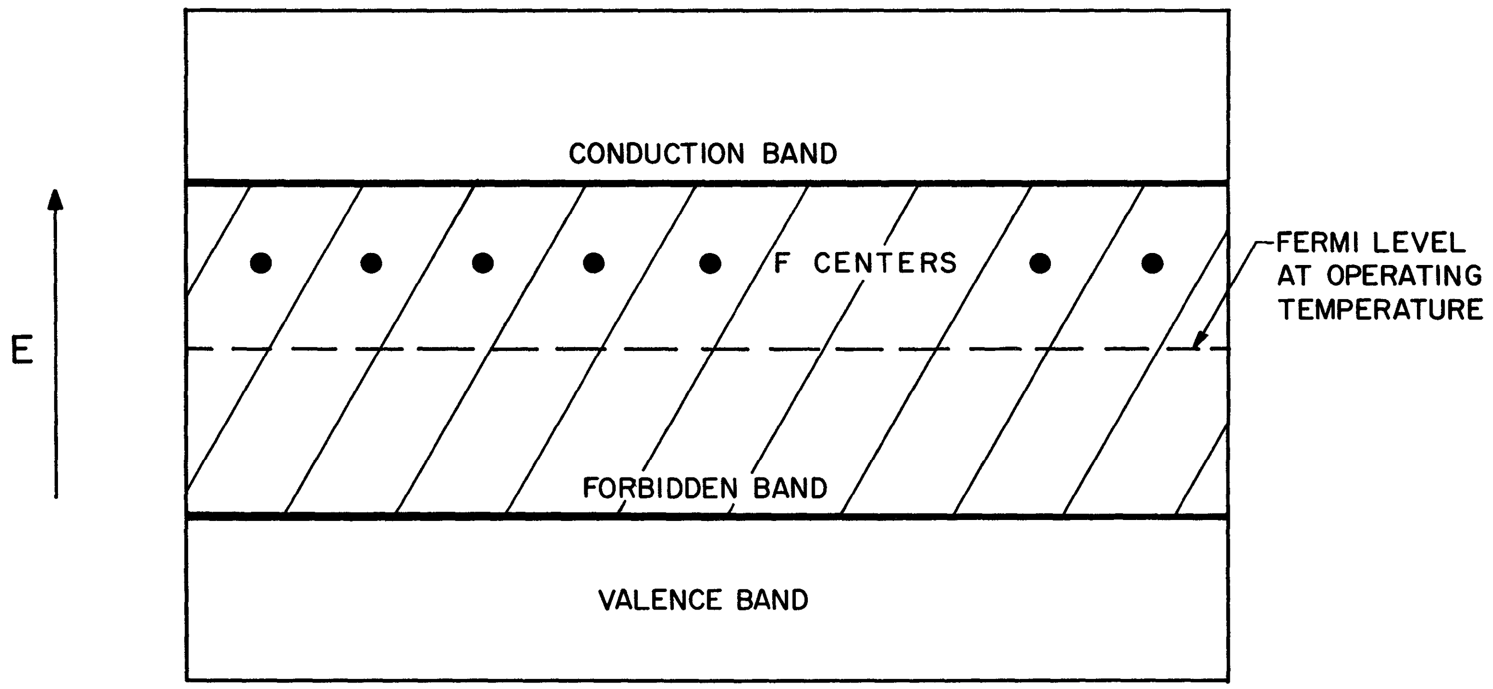


Figure 5. CRYSTAL BAND STRUCTURE

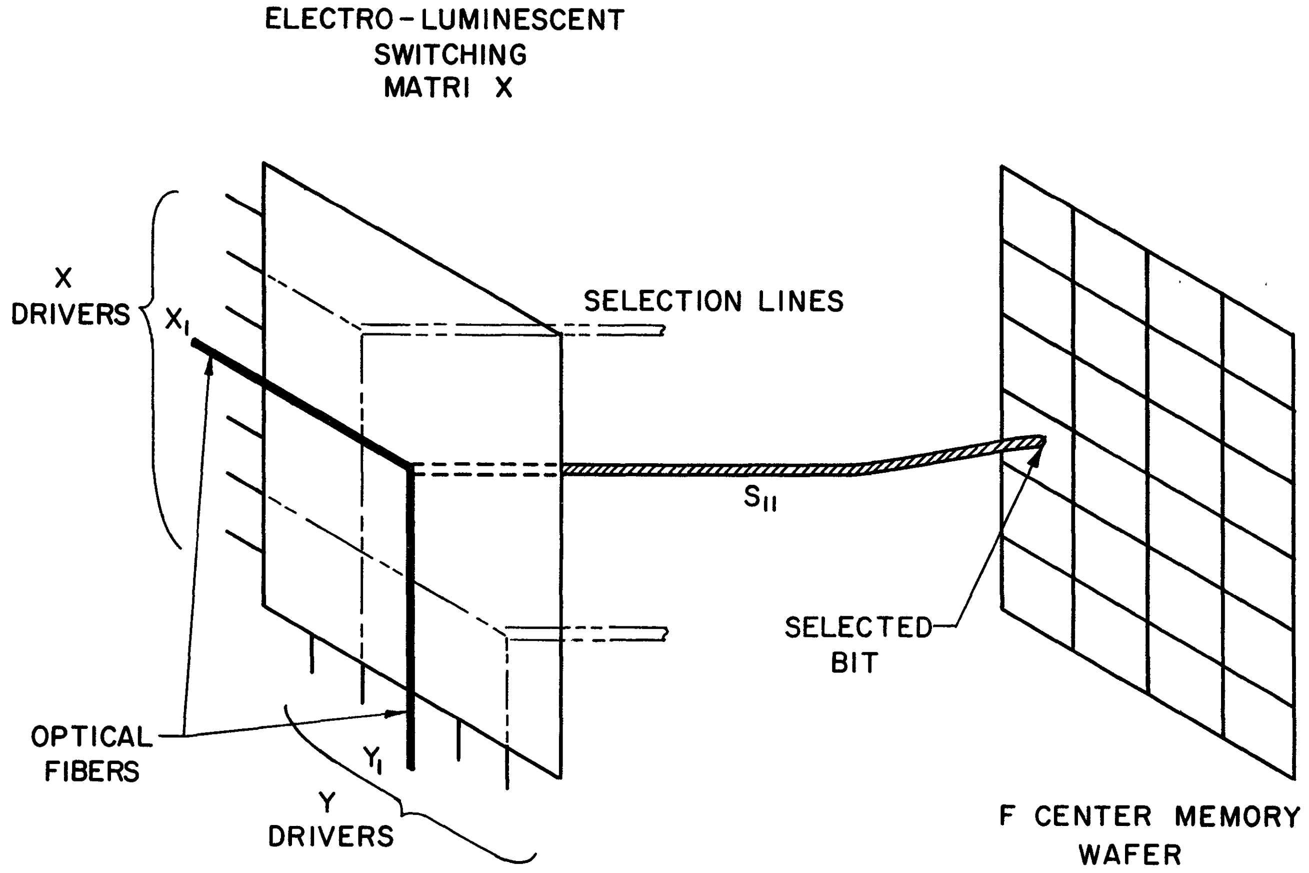


Figure 6. OPTICAL SELECTION OF A MEMORY CELL

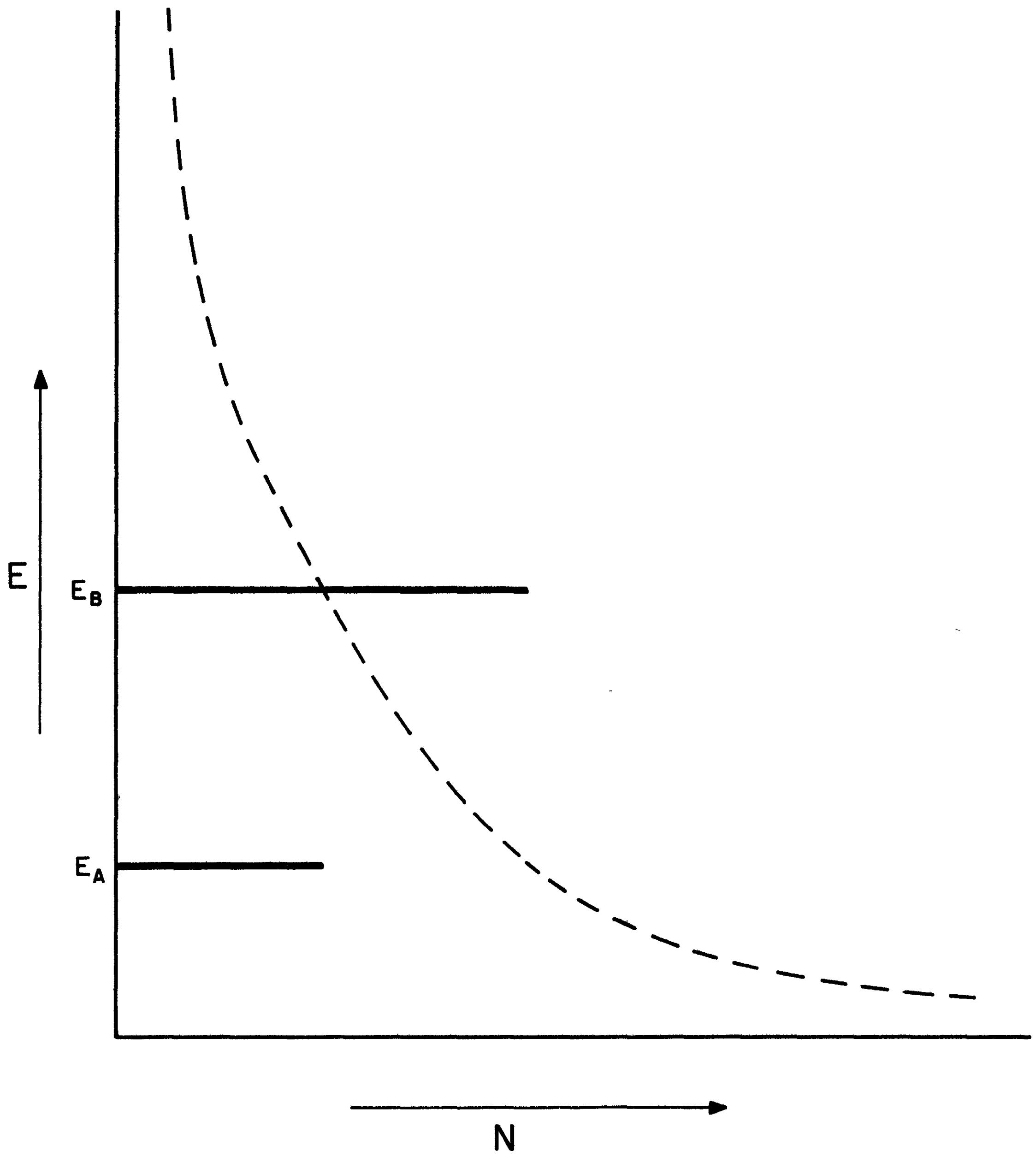


Figure 7. INVERTED POPULATION

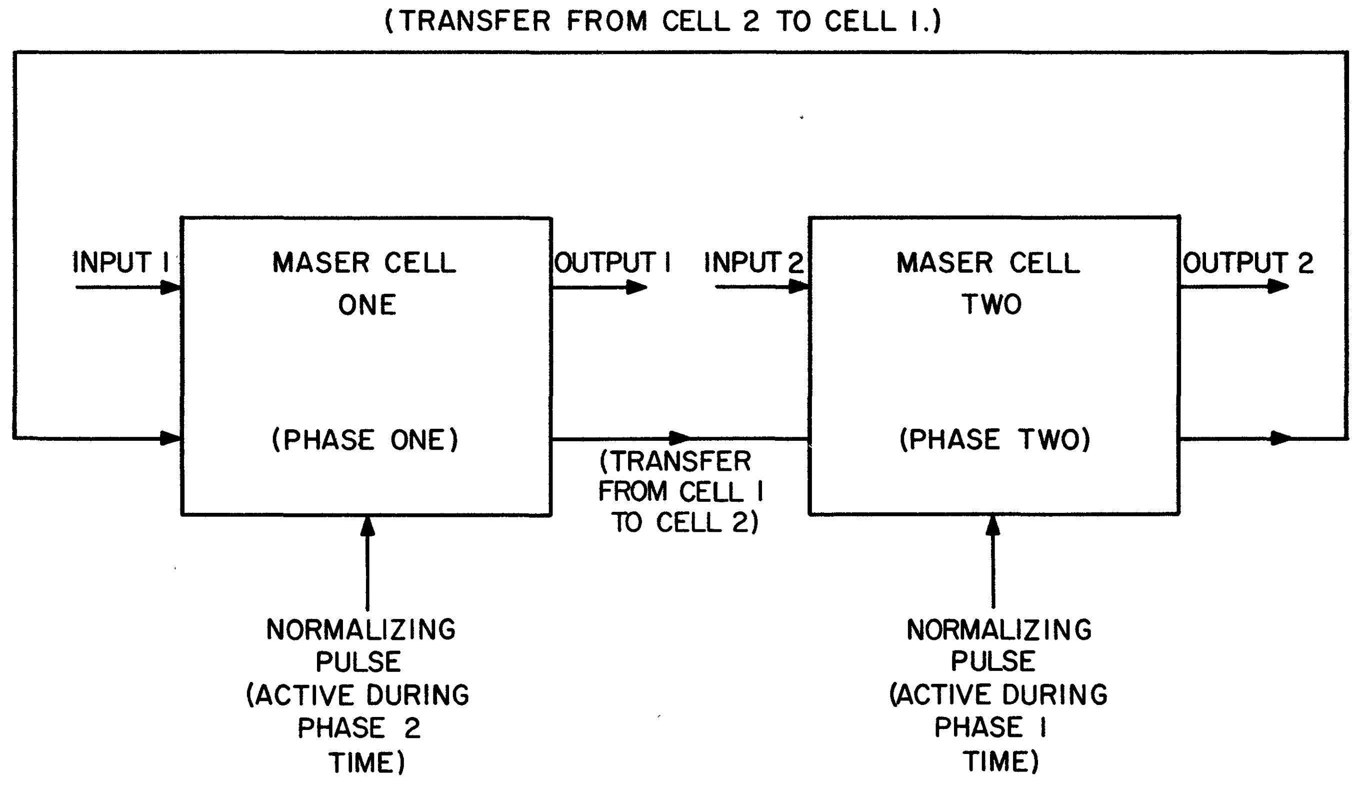

(TRANSFER FROM CELL 2 TO CELL 1.)
INPUT 1
MASER CELL ONE
(PHASE ONE)
OUTPUT 1
INPUT 2
MASER CELL TWO
(PHASE TWO)
OUTPUT 2
(TRANSFER FROM CELL 1 TO CELL 2)
NORMALIZING PULSE (ACTIVE DURING PHASE 2 TIME)
NORMALIZING PULSE (ACTIVE DURING PHASE 1 TIME)

Figure 8. CONTINUOUS STORAGE OF A BIT WITH TWO CELL MASER

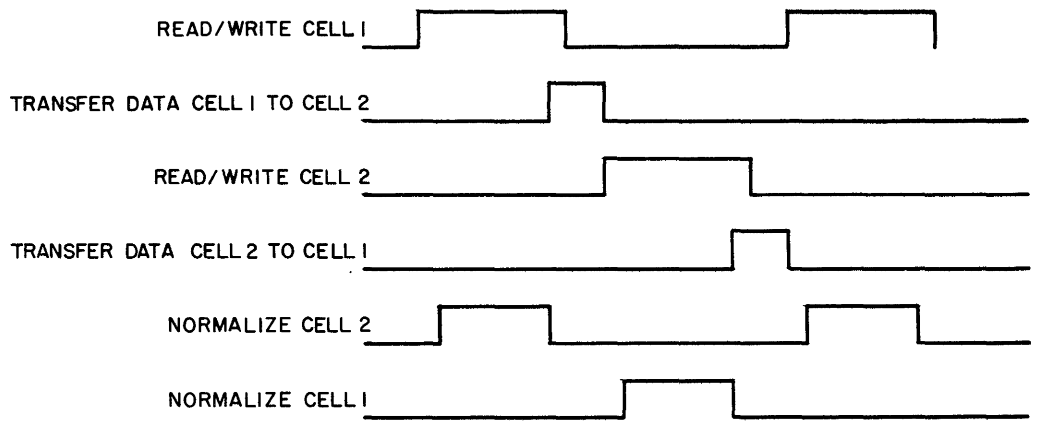


Figure 9. TIMING FOR CONTINUOUS STORAGE

BEAM 1	BEAM 2	RESULTANT BEAM
C_L	C_L	C_L
C_L	C_R	P
C_R	C_L	P
C_R	C_R	C_R

C_R = RIGHT CIRCULAR POLARIZED
C_L = LEFT CIRCULAR POLARIZED
P = PLANE POLARIZED

TABLE ONE
COMBINATION OF TWO POLARIZED BEAMS

OPTIMIZATION OF A RADAR IN ITS ENVIRONMENT BY GEESE* TECHNIQUES

Eugene L. Berger and Robert M. Taylor
Advance Systems Engineering
Defense Systems Department
General Electric Company
Syracuse, New York

SECTION I

Summary

One of the most complex problems in optimization of ground radars is the evaluation of the ground clutter and how it influences the operation of the radar. A fundamental limitation of ground radars is that the radar is unable to distinguish stationary targets in the presence of large amounts of random scatterers. The capability of a phase comparison radar to pick the stationary targets out of random scatterers has been evaluated on GEESE under a contract for Frankford Arsenal in Philadelphia, Pa. A model of Frankford Arsenal's radar was built on the GEESE analog computer facility and tested in single and multiple target situations in an environment consisting of random scatterers (clutter). This was accomplished by simulating mathematical characteristics of ground clutter and applying this as an input to the simulated model. This report deals with the GEESE application, an optimization of a general radar block diagram. The optimization is carried out under an arbitrary set of boundary conditions for the overall system.

A Discussion of GEESE

General Electric developed analog simulation techniques for evaluating entire electronic systems to meet the need for a faster, less expensive means for evaluating, analyzing and developing weapon systems.

In considering a system evaluation, there are four basic facts to remember:

1. Any signal can be generated and controlled easily.

2. All signals and waveforms can be recorded - this includes transients as well as steady state signals.

3. Feasibility studies can be undertaken on a simulated basis prior to development of system hardware.

4. No equipment need be procured for this preliminary investigation.

Thus GEESE permits the electronic system engineer to predict and optimize system performance.

GEESE techniques were pioneered at General Electric on the interference program associated with the development of the radio-command guidance system for the Air Force Atlas. A facility exists at General Electric's Defense Systems Department in Syracuse, New York, which is used to simulate all types of radar and communications systems and to evaluate the effects of ECM and mutual interference.

Purpose

The purpose of this study was to see if ground clutter could be simulated on the computer; and, as a check upon the simulation, the output of the clutter generator was to be compared to actual clutter. Also, the simulation of the radar was to be checked and compared to experimental data taken in the field by an actual radar which was used and designed by Frankford Arsenal, Philadelphia, Pa. Thus, if the simulation of both the clutter and the radar are correct, the results of this test should be the same as those obtained with the actual operating radar.

System Description

A radar can be analyzed by describing the transfer functions of the receiver and signal processing circuitry; thus, on the analog computer, a simulation of these transfer functions and signal processing circuitry yields the capability of analyzing all of the parameters of the radar. Second, the effects of ground clutter upon this radar can be determined by a simulation of the clutter environment. The ground clutter environment is the effect of trees, grass, bushes, and other moving objects upon the radar. In addition to these moving scatters, there are also fixed targets which are part of the ground clutter configuration. These are basically rocks, buildings, tree trunks, cliffs, ridges of earth, and other stationary man-made objects. The whole clutter environment, therefore, is quite complex in that it is composed of both random scatterers and fixed targets. It is the object of any ground radar to distinquish a particular stationary target in this complex clutter environment. The radar simulated is basically a monopulse radar. This means that the radar has two antennas and that simultaneous lobing occurs such that the phase of the target returns is the measure of the location of a given stationary target in azimuth, and the occurrence of the pulse is a measure of the range of the particular target. In this monopulse radar, there are two channels, a sum and a difference channel. The sum channel is the sum of all the instantaneous voltages arriving at the two antennas while the difference channel is the difference of the instantaneous signals arriving at the antennas. These signals are then bandpassed in an IF strip and video detected. The

* General Electric Electronic Systems Evaluator

video signal is integrated and processed by boxcarring. These boxcarred signals are then subtracted and the final presentation is this subtraction of boxcars. When a given fixed target is on boresight and there is no clutter, then a maximum will occur in the sum channel when a minimum occurs in the difference channel. If these two signals are then video detected and boxcarred, the sum channel boxcars will then be a maximum while the difference channel boxcars will be a minimum. The subtraction of these two boxcars will yield positive boxcars. If there is no target, only random scatters, then the sum and difference channels will have independent random noise signals and the boxcar outputs of each channel will be both positive and negative. The subtraction of these two signals will give a boxcar output which varies around zero both positive and negative; thus, in this system, a positive boxcar output is an indication of a target.

Area of Investigation

The radar was investigated in fundamental form in different clutter environments. A range of target to clutter power ratios was investigated by varying the amount of clutter input. The clutter was characterized by wind speed and the particular center frequency of the radar. The radar was also investigated in a fundamental form without any clutter by supplying only target information to the radar. The target was then moved in azimuth across the antenna pattern of the radar. This determined the sum and difference patterns of the respective channels of the radar. The radar again was investigated in its fundamental form when two targets were the only input to the radar. By varying the two targets in azimuth across the antenna beamwidth, the interaction in either the sum or difference channel could be noted.

SECTION II

Simulation

GEESE Simulation

The analog computer simulation of this radar is a scale model of the given system and not an analog. In all systems simulated on GEESE, only a scaling of frequencies occurs with gain relationships, voltages, and currents remaining in the same order of magnitude. More important, certain non-linearities are taken into account. Waveforms observed at the terminals of system elements are identical to those in the actual system except for time scaling. On the whole, the simulation is highly idealized compared to the actual system. The amplifiers and integrators are linear over their entire range, and system elements such as a linear detector are extremely linear. The input frequencies to the analog computer can be made to have the same order of stability as those experienced in the actual system. Noise inputs are purely Gaussian in amplitude and white in power density. The signal levels put into the analog computer were held constant without the action of an AGC loop, and the input frequencies were held constant without an AFC loop.

The Simulation of Ground Clutter

Actual clutter is characterized by power frequency spectrum and a correlation function. The clutter generator simulates these characteristics as frequencies scaled compatibly with the radar parameters under consideration. The power frequency spectrum and the correlation function simulated were determined from a study of actual clutter. In a situation involving random scatterers (clutter) and a single stationary target, the composite return has a Rayleigh distribution. As the target level increases, the probability density function of the signal power approaches a Gaussian distribution. The mean of this Gaussian probability density function is the signal power of the target. Appendix 1 presents a mathematical analysis of clutter.[1]

The clutter supplied to the simulated radar comes from a clutter generator which implements this mathematical model of ground clutter. This generator implements and incorporates some of the features particular to the radar in this study and includes the flexibility required to simulate a wide variety of environmental conditions. The clutter generator, in general, provides two basic components of clutter. The first is random scatterers, which consist of the returns from leaves and grass in various wind speeds up to gale winds. The second main component of clutter is stationary targets. These targets are representative of dense woods, rocks, boulders, and other ground environments which are stationary. The clutter generator can position these stationary targets in range and in return power. Also, the clutter generator can position the stationary targets in azimuth. Another capability of the clutter generator is to simulate amplitude and frequency of range jitter. This can correspond to frequency instability in the transmitter or to the motion of a given fixed target.

The block diagram shown in Figure 1 represents the signal processing which is carried on in the clutter generator. Each channel receives equivalent power returns from a collection of random scatters, two parasitic targets varying in range, and one fixed target. The random scatters have the characteristics of the purely theoretical random scatters in that the probability distribution of the amplitudes is a Rayleigh distribution and correlation function is equivalent to that observed in experimental data. Also, the amplitude frequency spectrum of the random scatterers is based upon the experimental findings of clutter returns at different wind speeds for various RF center frequencies.

In the block diagram, four independent Gaussian noise sources are used to modulate two carrier components in phase and quadrature. These components are added and then supplied to both the sum and difference channels. The output of the sum amplifier in the block diagram is then Rayleigh distributed in amplitude and has a power frequency spectrum corresponding to a particular wave length of the radar and the wind speed of the environment. When a main target is introduced into the summing amplifier in the block diagram, the output is now Gaussian distributed in amplitude having a mean about the average amplitude return from the steady target. By varying the main target in amplitude and phase, so as to simulate frequency instability or motion of the target, the output of the summing amplifier in the block diagram changes from the Rayleigh

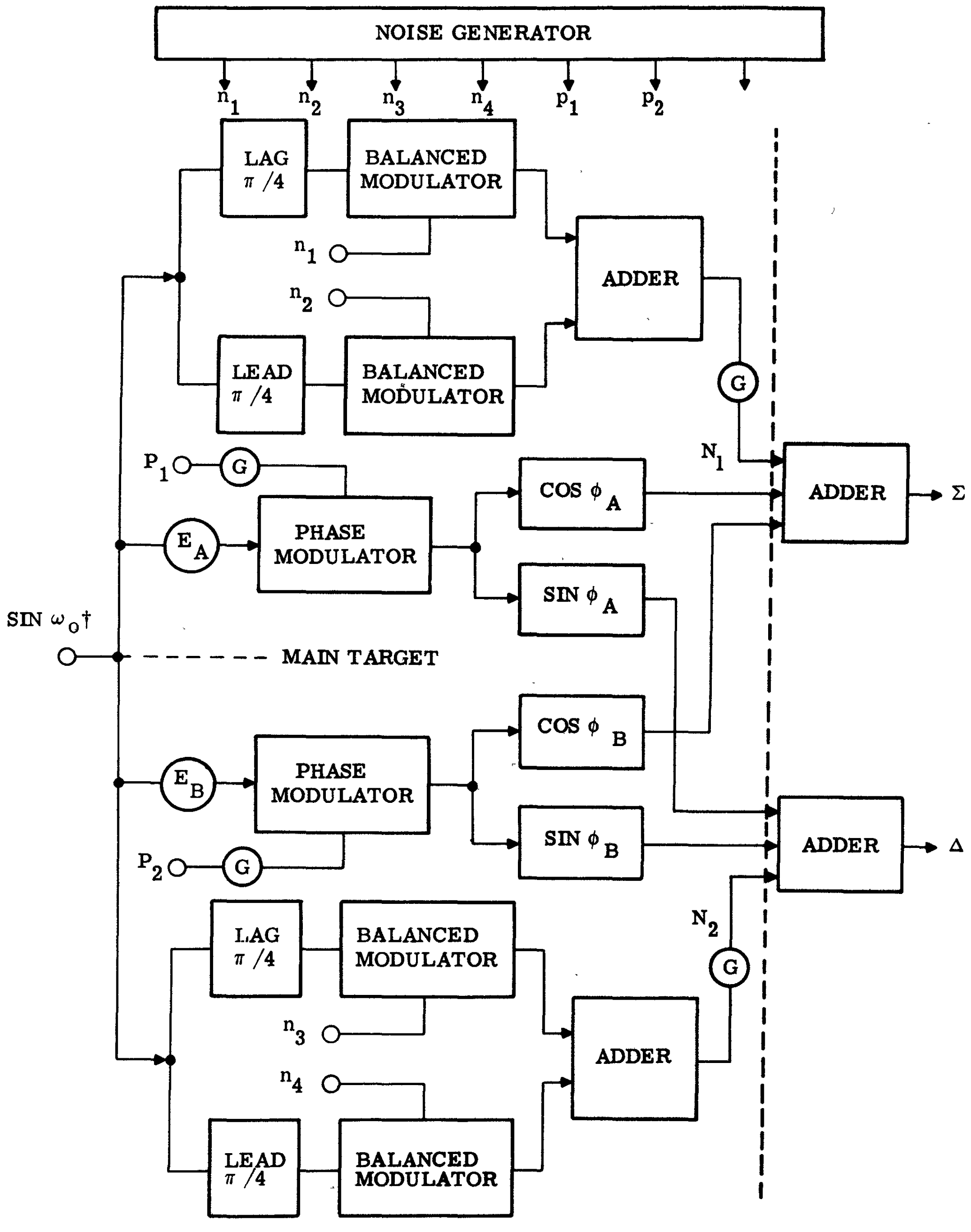


Figure 1. Block Diagram, Signal Processing of a 400-CPS Carrier

distribution in amplitude to a particular Gaussian distribution in amplitude, therefore, describing the theoretical probability distributions for signal plus noise where the signal is also varying in amplitude and phase. The parasitic targets in the block diagram are introduced through the phase modulator and resolving potentiometers which respectively jitter the targets in range and describe their positions in azimuth.

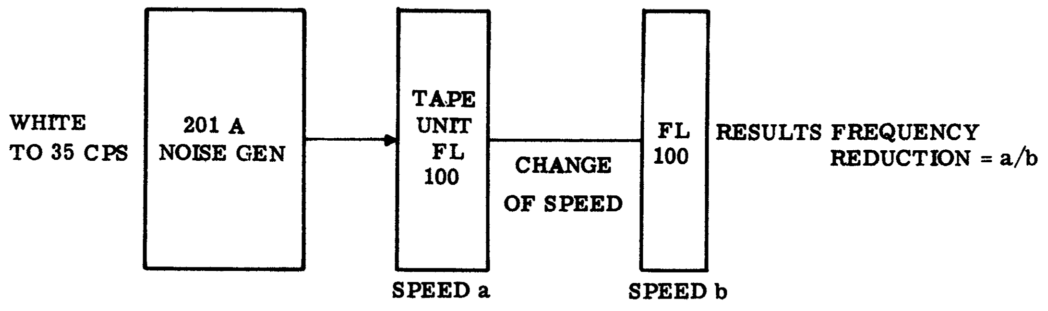
This whole scheme is based on the instantaneous sum and difference signals appearing in a monopulse radar. The sum phasor is multiplied by the cosine of the respective electrical angles off boresight; and, in the difference channel, the instantaneous phasor is multiplied by the sine of the respective electrical angles off boresight. These equations follow.

$$e_{sum} = \mathrm{Re}\left\{ 2 \left[\tilde{E}_A \cos \phi_A + \tilde{E}_B \cos \phi_B \right] e^{jw_o t} \right\}$$

$$e_{difference} = \mathrm{Re}\left\{ 2 \left[\tilde{E}_A \sin \phi_A + \tilde{E}_B \sin \phi_B \right] e^{jw_o t} \right\}$$

The noise generator in the block diagram is essentially a tape unit plus two low-pass filters which give the amplitude versus frequency spectrum of the noise and the correlation function of ground clutter. This is illustrated in Figure 2 which shows how the frequency spectrum is set and the correlation function is obtained by two low-pass filters. Appendix 2 describes how the correlation function is achieved by two low-pass filters. Knowing the frequency spectrum coming out of the tape unit and the power density of this noise, the output power of the two low-pass filters can be calculated by the following equation which has been derived in Appendix 3.

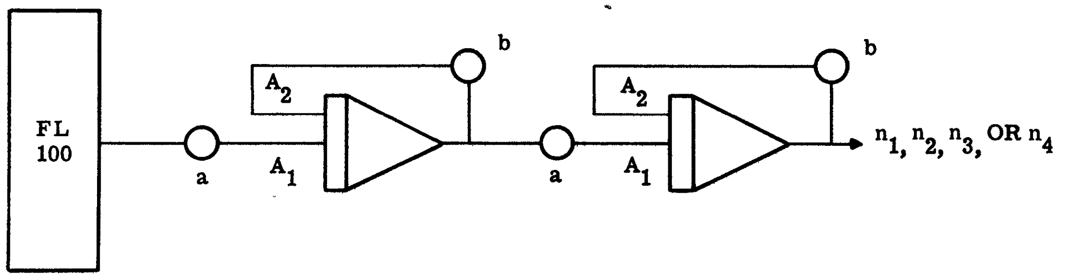
$$\therefore \overline{x_2^{\,2}} = \frac{a^2 A^2}{w_1^{\,2}\, 2} \left[\frac{1}{1 + (w_o/w_1)^2} + \frac{w_1}{w_o} \mathrm{Tan}^{-1} \frac{w_o}{w_1} \right] \overline{x_o^{\,2}}$$



if a = 60 in. /sec $\quad \frac{a}{b} = 8$

b = 7-1/2 in. /sec

$\therefore$ The new spectrum is white to $\frac{35}{8} = 4.37$ cps



Various values of a and b are determined by wind speed and the rms output level.

Figure 2. Sequence of Operation

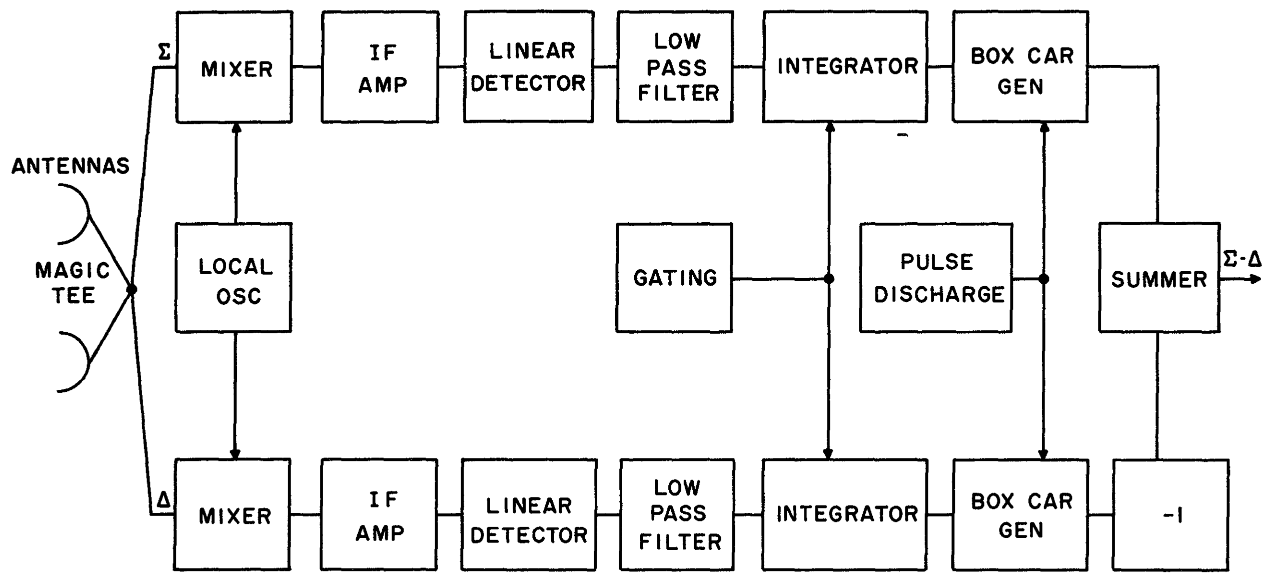


Figure 3. Radar Receiver Block Diagram

General Description of the Simulated Radar

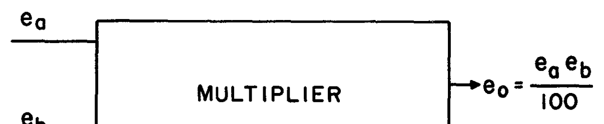
The block diagram in Figure 3 is that of the radar receiver simulated on the GEESE analog computer. The block diagram consists of two antennas, a magic T, a first detector, a second detector, and a third detector. After the third detector, there is a boxcar generating circuit and a subtractor circuit which subtracts the sum and difference boxcars.

The magic T is simulated in the clutter generator so that the sum and difference of the antenna patterns are taken directly from the clutter generator. The first detector, which is a mixer converter, is simulated on the computer by a multiplier. This is shown in Figure 4. If the inputs are E_a and E_b and $E_a = A_1 \sin w_o T$ and $E_b = A_2 \sin w_1 T$, then the output is of the form

$$\frac{E_a E_b}{100} = \frac{A_1 A_2}{200} \cos (w_o - w_1) T - \frac{A_1 A_2}{200} \cos (w_o + w_1) T$$

This acts as conversion from RF to IF frequencies. The second detector is a bandpass filter which is normally referred to as an IF strip. The IF strip is centered such that only the sum frequency is passed through the IF filter. This IF filter also has the characteristics of the original radar in that it has a bandwidth equivalent to the IF of the experimental radar, and it also has a center frequency equivalent to the center of the experimental radar. The transfer function is identical to that of the experimental radar in that the simulated IF is a staggered tuned triplet. In Figure 5 is the simulation of one of the IF amplifiers. This filter has the transfer function:

$$\frac{K_1 s}{s^2 K_2 + s K_3 + K_4}$$

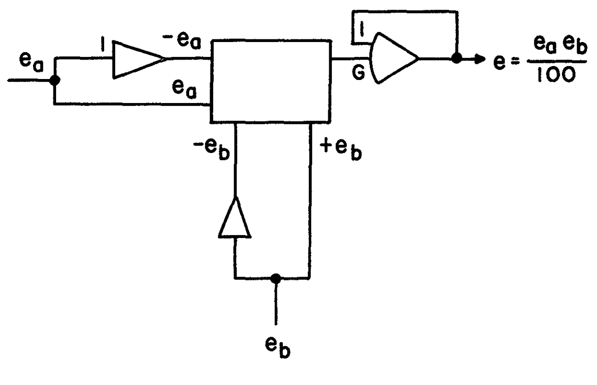


$e_a = a_1 \text{ SIN } \omega_0 t$

$e_b = a_2 \text{ SIN } \omega_1 t$

$$\frac{e_a e_b}{100} = \frac{a_1 a_2}{200} \text{COS} (\omega_0 - \omega_1) t - \frac{a_1 a_2}{200} \text{COS} (\omega_0 + \omega_0) t$$

SIMULATOR CIRCUIT



COMPUTER DIAGRAM

Figure 4. GEESE Equivalent Circuit of First Detector (Mixer)

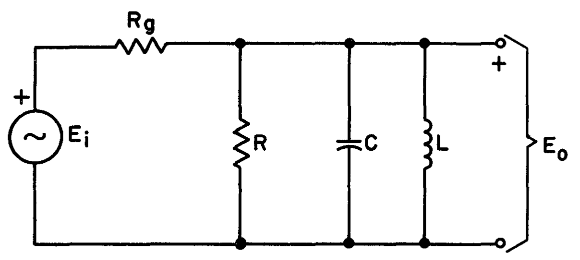


Figure 5A. Narrow Band Pass Filter

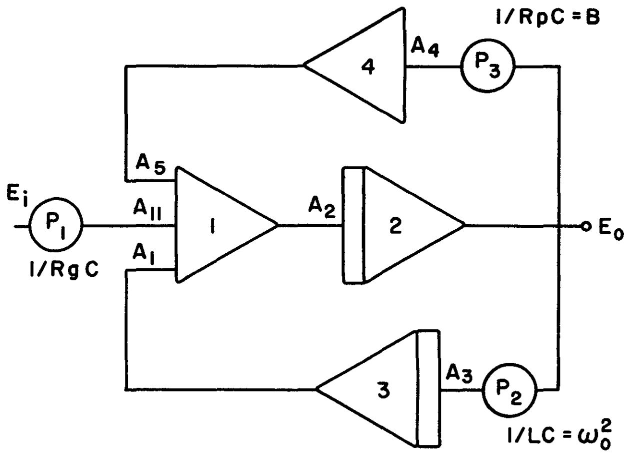


Figure 5B. GEESE Equivalent Circuit

Figure 5.

This computer simulation is the exact equivalent to a single tuned circuit. Staggered pairs and triplets can be formed by cascading several of these. In Figure 7 is the simulation of the stagger tuned triplet simulated on the computer. The first IF amplifier in this stagger tuned triplet has a relative gain of 2 and a bandwidth of 10 cycles. The second bandpass filter has a relative gain of one and has a bandwidth of 20 cycles per second. The third is identical to the first with the exception that it is at a higher center frequency. The resultant magnitude plot of these three staggered tuned circuits is illustrated in Figure 7.

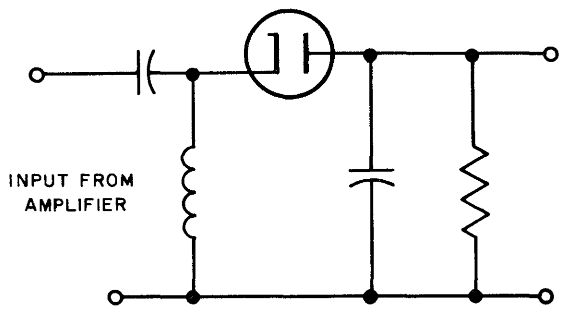
The third detector is a linear detector and a low-pass filter. The linear detector has the relationship $e_{in} k = i_b$ which is the normal characteristic curve for a perfect linear detector. The third detector is shown in Figure 6. The low-pass filter has the transfer function of:

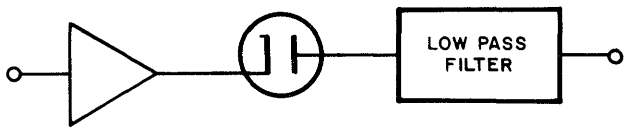
$$H(s) = \frac{1}{s + \frac{1}{RC}} = \frac{1}{s + w_1} .$$

The bandwidth of this low-pass filter is set on the computer to that bandwidth corresponding to the third detector in the actual radar. The computer simulation of the low-pass filter can be seen in Figure 7 which is a complete GEESE model of the simulated radar.

The boxcar generator consists of an active integrator which allows integration only during the sampling period or gating period. After the gating period, the input is switched to zero so that the output of the integrator remains constant until sometime later when it is desired to discharge the integrator and to initiate the integrating again. This is all shown in Figure 7 where the circuit discharge is made through the resistances Q 09 and Q 28 at the discharge occurring time.

Table 1 gives the IF center frequencies, the RF frequency, the IF bandwidth, and other characteristics of a representative radar. Corresponding to these parameters of the radar are the computer scaled parameters. The RF center frequency was changed to 400 cycles per second since there was no information in the carrier frequency; therefore, it could be changed to any convenient carrier. The other parameters were scaled by 10^{-6} with the exception of the clutter frequen-





TYPICAL RADAR THIRD DETECTOR

COMPUTER EQUIVALENT CIRCUIT

Figure 6. Radar Third Detector and Equivalent Circuit

cies and the pulse repetition frequency. These two parameters were scaled to the 10^{-3} so that the computing time would not be so extremely long had it been scaled to 10^{-6}. Figure 7 includes all the filter characteristics, the boxcar characteristics, etc. The output of the two boxcar generators are subtracted from each other in an operational amplifier after the inversion of the difference channel signal. Thus, a representative radar receiver is simulated on the computer, taking into account all of the bandwidth characteristics and the transfer characteristics for each of the blocks in the radar.

Data Recording Processes

Two types of data recording are used throughout this study. The primary method is "A" scope presentations photographed for varying periods of time. The photographic process provides an accurate simulation of real time "A" scope integrations. The majority of the photographs in Section III were exposed for a period of 5 minutes. This corresponds to approximately 1200 traces, equivalent to 1/4 of a second in the real system. Twelve hundred traces are sufficient[2] to produce a stationary presentation, i.e., a reliable signal to clutter ratio.

The second method of data recording and display is eight channel oscillograph recordings. This type of data recording process enables us to examine individual traces on a continuous basis, that is, a non-integrated situation. In addition, this method enables us to monitor several points in the system to insure correct functioning of all stages of the simulated system. Data recorded in this fashion can be statistically analyzed by an examination of individual output traces in compilation of an overall statistical result for the system.

TABLE I

Radar Characteristics	Scale Factor	Computer Simulation
RF Center Frequency 35,000 mc/sec	----	400 cps
Pulse Length 0.06 μsec	10^{6}	0.06 secs
IF Bandwidth 20 mc	10^{-6}	20 cycles
IF Center Frequency 60 mc/sec	10^{-6}	60 cps
Gating Width 0.06 μsec	10^{6}	0.06 secs
Length of a Boxcar 250 milli/sec	1	250 milli/sec
Discharge Time 5 milli/sec	1	5 milli/sec
Pulse Repetition Frequency 4,000 cps	10^{-3}	4 cps
Video Bandwidth 15 mc	10^{-6}	15 cps
Clutter Frequencies X cps	10^{-3}	X 10^{-3} cps

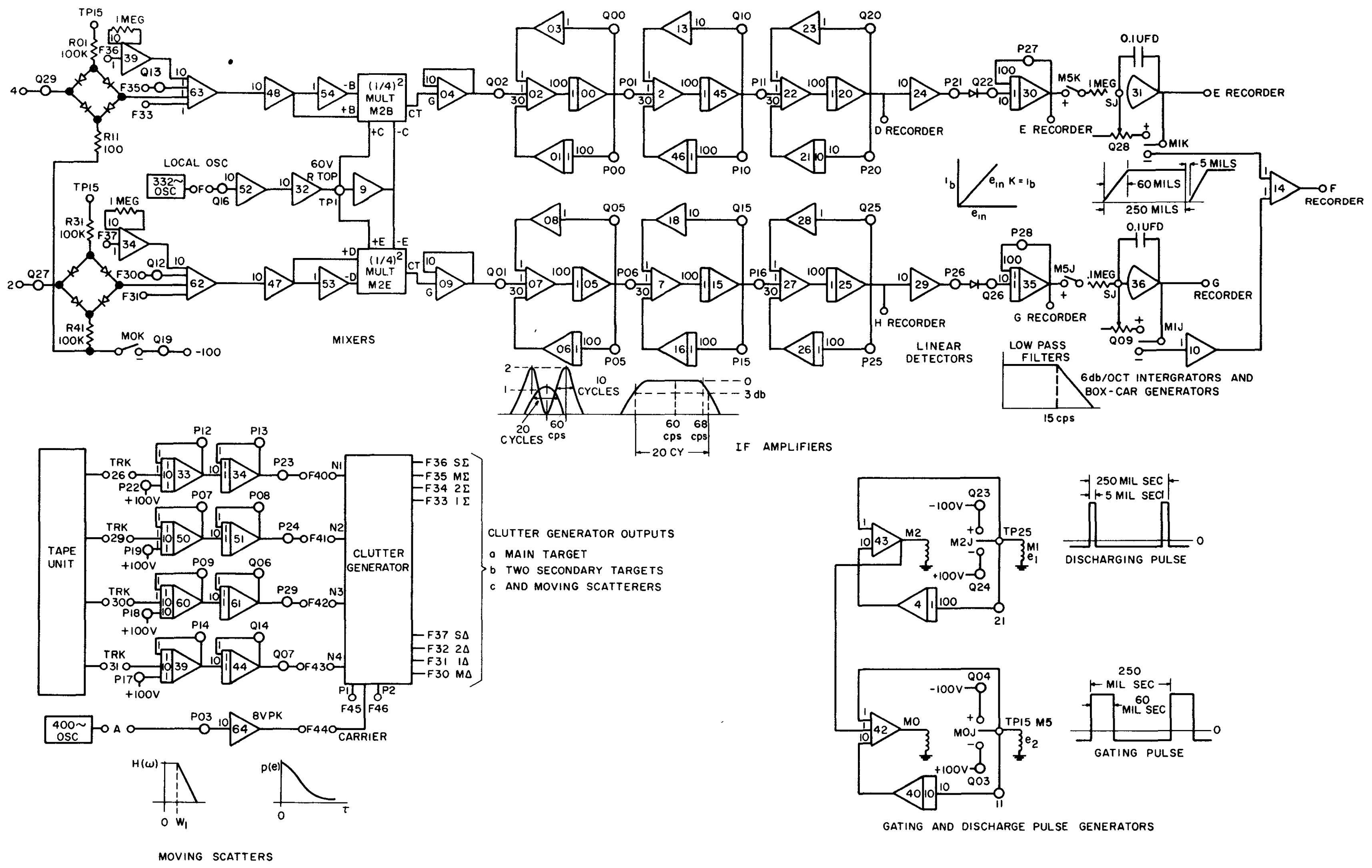


Figure 7. Complete GEESE Model of Radar Receiver

SECTION III

Results

The Investigation of Different Characteristics of the Radar

The clutter environment was applied to the simulated radar to investigate different characteristics of the radar. In the condition investigated, there was one stationary target and random scatterers. The ability of the radar to detect the stationary target was investigated for different m^2's where the m^2 is the ratio of steady power returns to random power returns. The results of this run are shown in Figure 8. In this figure, the output boxcars, which are the sum minus the difference boxcars, are shown for $m^2 = 1.32$, 2, and 4 when the main target is on boresight. The results of this show that for $m^2 = 1.32$ that the probability of detecting a target is 70 percent. The probability of detecting a target for $m^2 = 2$ is 95 percent. The probability of detecting a target for $m^2 = 4$ is 100 percent, so that on any m^2 above 4, the probability of detecting a target is always 100 percent. It can also be noted from Figure 9, which is the complete record for $m^2 = 1.32$, that the boxcars of the difference channel are that of a Rayleigh distribution in amplitude. The output of the boxcars in the sum channel are some Gaussian distribution with a mean about that of the amplitude of the target signal. This in turn verifies the output of the clutter generator to be exactly what it was designed to be. The other records in this figure are the sum and difference IF signals. These signals are essentially CW signals, but the results out of the boxcar generators are the same as they would have been had the inputs been pulsed. This is due to the fact that the boxcar generator is sampling the IF signals and integrating them only during a period of time equivalent to the time occurrence of a pulse. The only difference that can be said to exist if the inputs were pulsed is that the amplitudes of the sum and difference boxcars would be smaller due to the fact that some of the energy would have been lost due to filtering in the IF strip. An actual pulse input has been put into the computer to provide a range gated "A" scope presentation. This is shown in Figure 10.

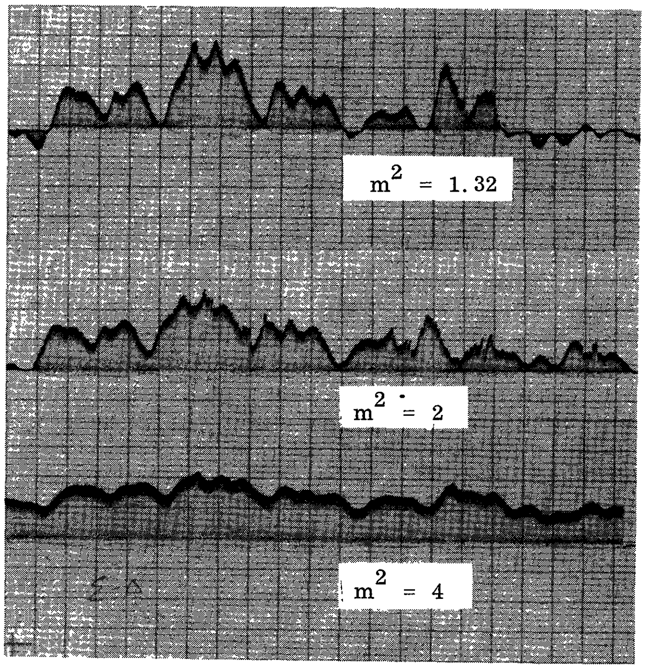


Figure 8. Single Target on Boresight in Clutter

The second investigation was to determine the system performance in the presence of two stationary targets of different sizes. Figure 11 is a series of recordings taken on the computer to determine if any interactions occur between the sum and difference channels. The strong target had twice the power of that of the small target, and the small target was jittered in phase so that the envelope of the sum and difference channels could be observed. The two targets were separated by 30 electrical degrees in azimuth. The frequency of the phase jitter on the small target was 0.3 cycle per second with a total phase deviation of 10 π radians. The different recordings are for different locations of the target and the parasitic. In all of the recordings, the main target is 30 electrical degrees to the left of the parasitic. It can be observed from the figure that the difference channel obtains its minimum when the stronger target is on boresight or its electrical position in azimuth is zero.

Figure 12 is an "A" scope presentation of two targets appearing at slightly different ranges, with the radar scans across the targets. The targets are being jittered in phase so as to simulate frequency instability or target motion.

It can be noted from these figures that the behavior of the simulated radar corresponds to that of actual radars. In fact, comparative results have shown the correspondence to be quite satisfactory for various specific situations.

At this point, optimization of the particular radar may begin. Parameters such as bandwidths, center frequencies, repetition frequency and pulse widths can be varied easily. In addition, specific antenna patterns can be investigated. This has been accomplished for a specific radar with results satisfactory to both the system engineers and the circuit designers.

As an example the video bandwidth in this report was found to be optimum at 12 cps instead of 15 cps. In addition to empirical methods of parameter optimization, analytical optimization techniques can be evaluated using these analog methods.

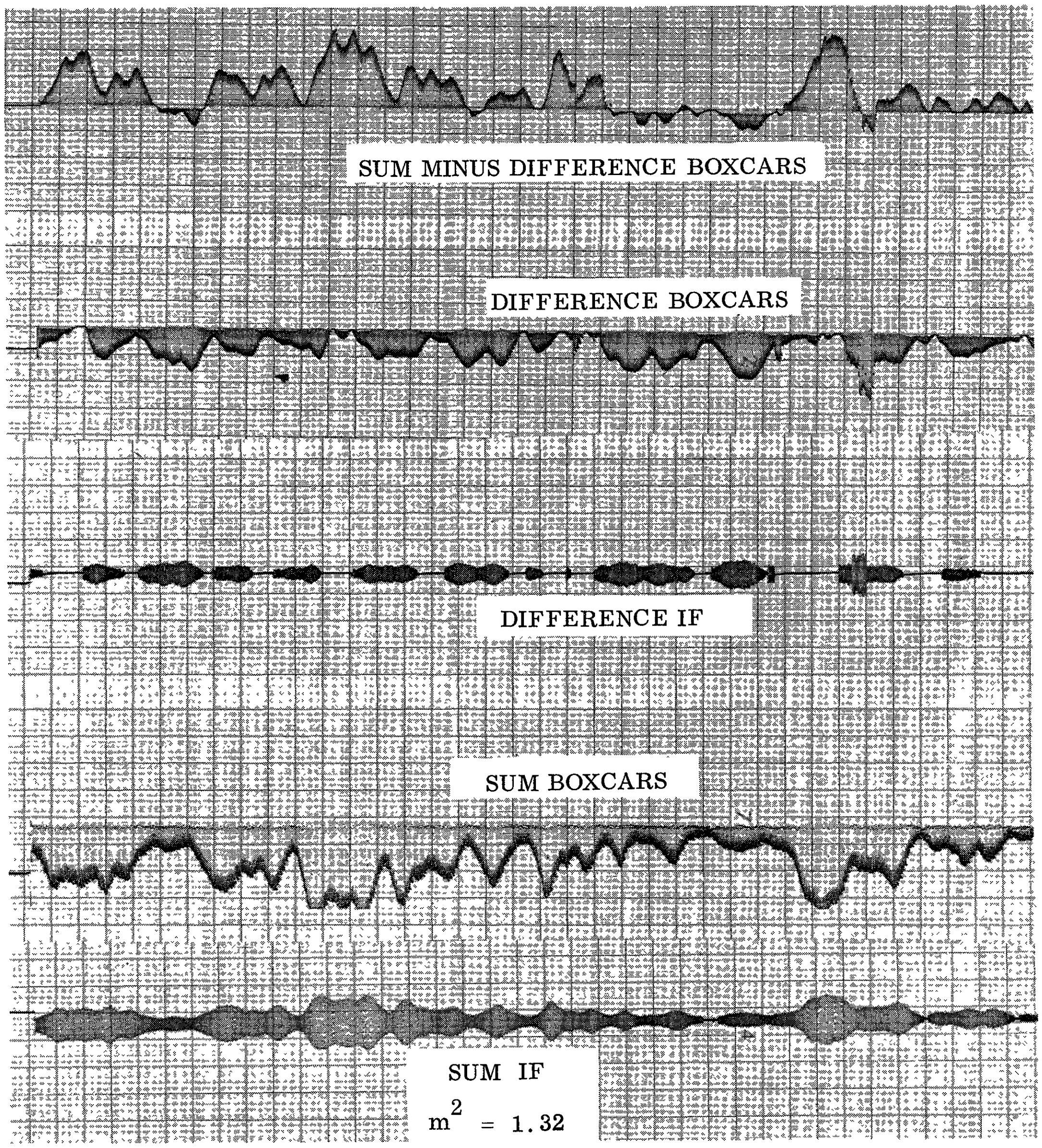


Figure 9. Single Target on Boresight in Clutter

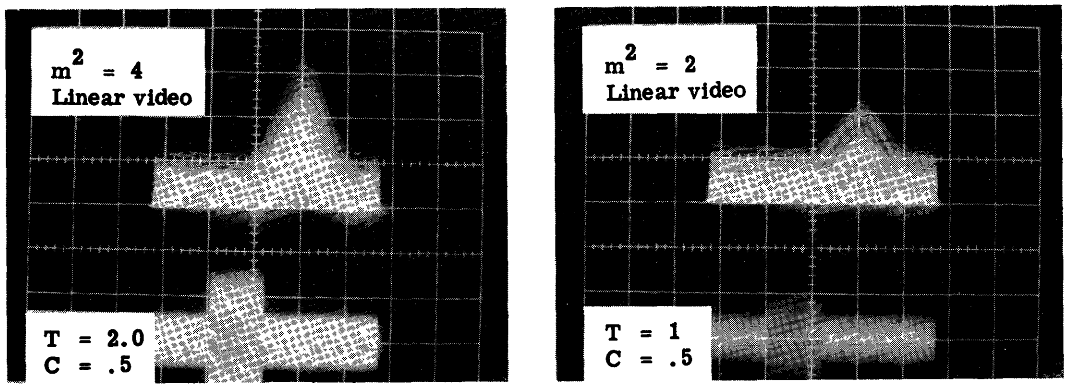


Figure 10. Single Target on Boresight in Clutter

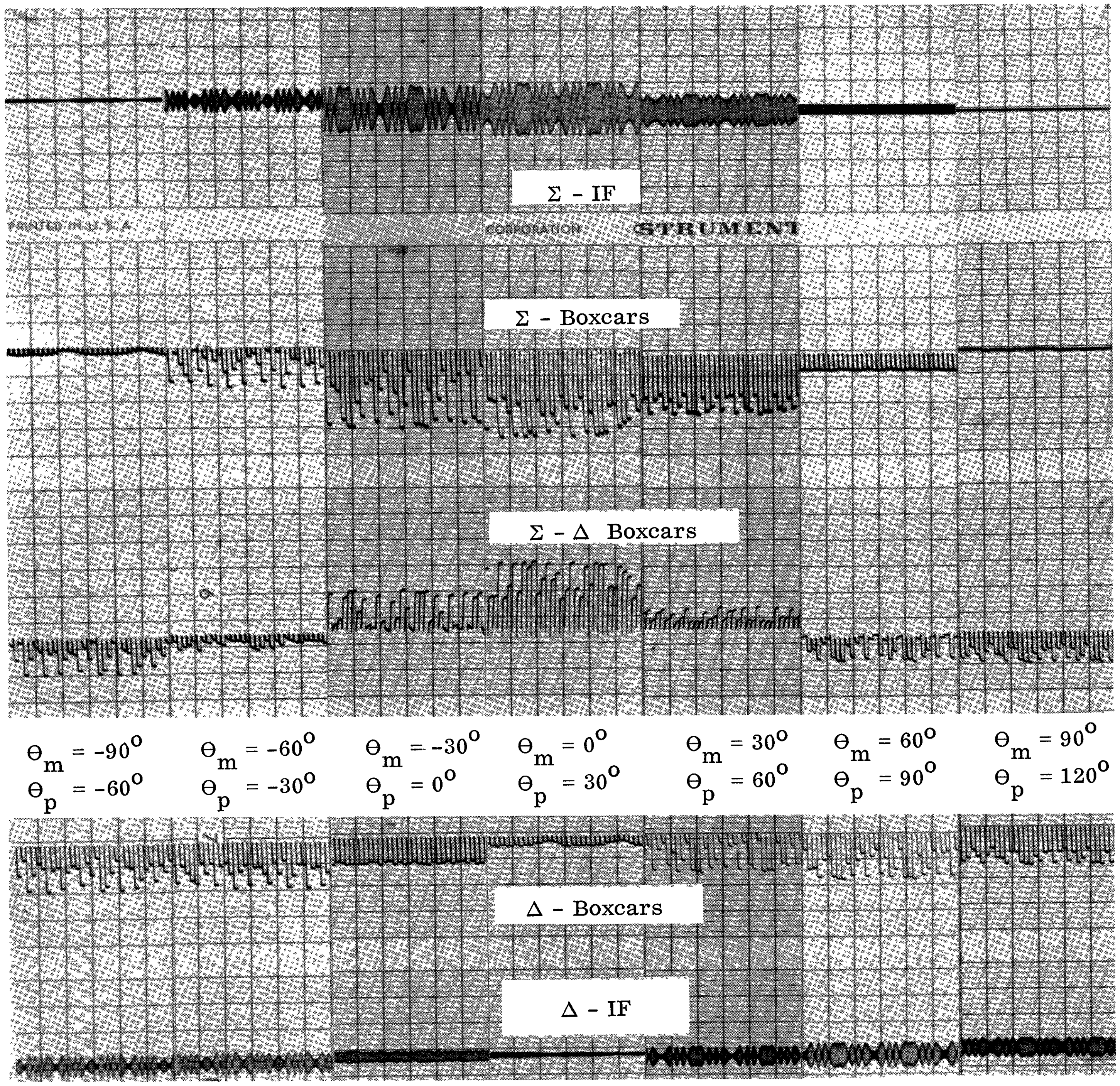


Figure 11. Two Targets Being Scanned in Azimuth

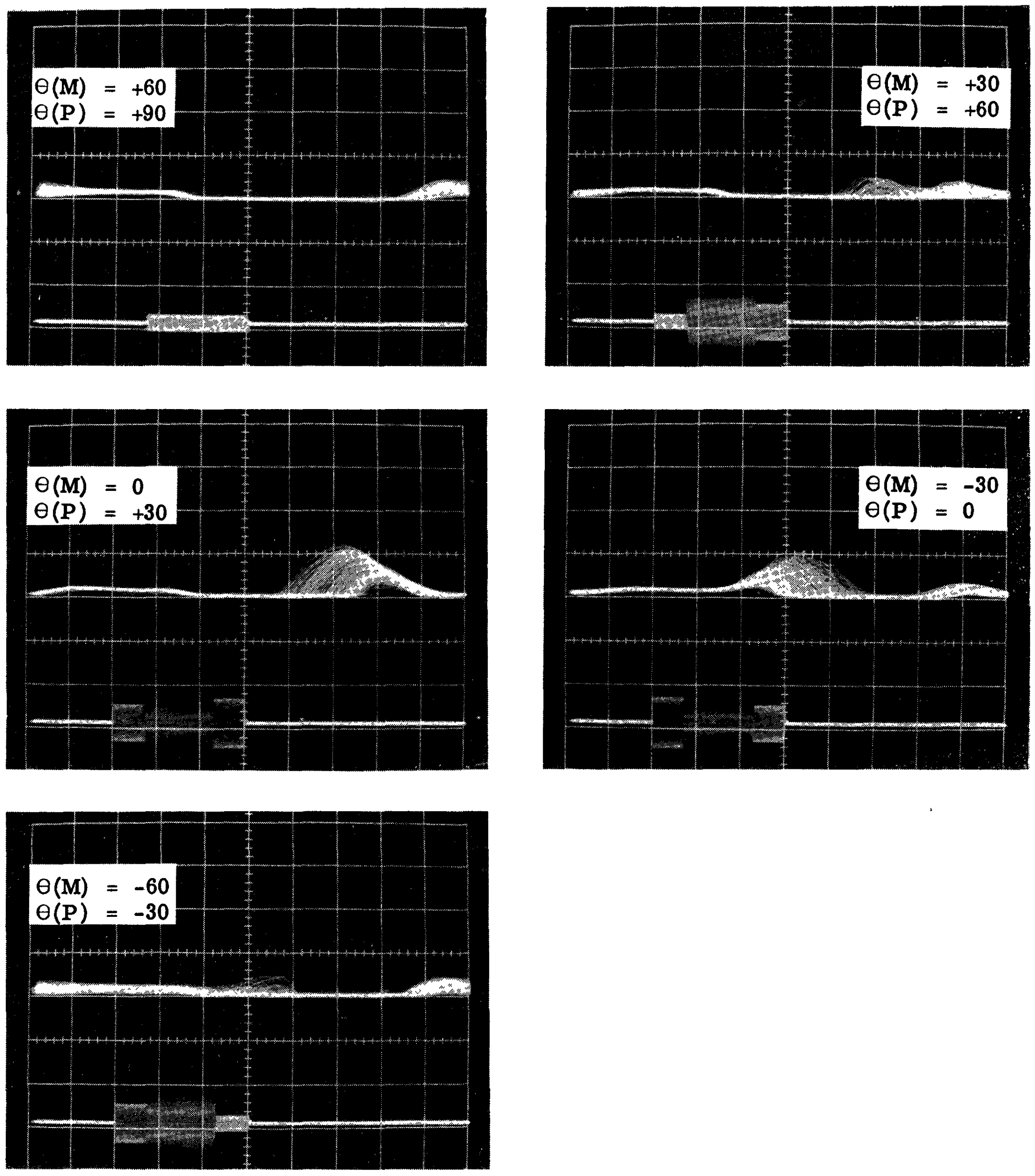


Figure 12. Two Targets Being Scanned in Azimuth, No Clutter

APPENDIX 1

Mathematical Analysis of Random Scatters

Description of Noise Envelope

Consider some of the statistical features about the envelope and phase of a random noise after passage through a narrow band filter. The frequency spread of the noise is seen to be small compared to the center frequency of w_c of the narrow band filter. If z(t) denotes the output noise record, then z(t) is equal to R(t) cos $w_c t$ - ϕ(t) with the envelope R(t). The phase angles ϕ(t) are allowing varying variables of time relative to oscillations of angular frequency w_c. From the previous expression z(t) can be represented as the sum of sines and cosines z(t) equal x(t) and the cosine of $w_t t$ - y(t) sin $\phi_c t$ where x(t) = r(t) cos ϕ(t) and y(t) = r(t) sin ϕ(t). This equivalent representation indicates the following relationships that $R^2 = x^2(t) - y^2(t)$ and Tan $\phi(t) = \frac{y(t)}{x(t)}$.

Suppose that the joint probability density function p(x, y) is known. Then the joint probability density function p (R, ϕ) can be found from the following equation:

$$p(x,y)\ dxdy = p(R\cos\phi,\ R\sin\phi)\ R\ dRd\phi.$$

Since the element of area dxdy in the x, y plane corresponds to the element of area of R dRdϕ in the R, ϕ plane, let Q(R, ϕ) = R p(R cos ϕ, R sin ϕ). Then p(x, y) dxdy = Q(R, ϕ) dRdϕ. Now the probability density function and the noise envelope Rϕ(t) alone is obtained by the sum of all phase angles and is

$$Q_1(R) = \int_0^{\beta\pi} Q(R,\phi)\ d\phi.$$

While the probability density function of the phase angle ϕ(t) alone is obtained by summing over all possible R and is

$$Q_2(\phi) = \int_0^{oo} Q(R,\phi)\ dR.$$

If x(t) and y(t) are each normally distributed about zero and mean squared values of x(t) and y(t) are equal and equal to the mean square value in c(t), also if the cross correlation function between x(t) and y(t) so that x(t) and y(t) are independent random variables, hence x(t) and y(t) can be expressed as sums of normal variables. We conclude that their joint probability density function will be two dimensional normal distribution of the form

$$p(x,Y) = p(x)\ p(y) = \frac{1e^{-\frac{x^2+y^2}{2\sigma^2}}}{2\pi\ \sigma^2}$$

During the integration described previously, it follows a

$$Q_1(R) = \frac{Re^{-\frac{R^2}{2\sigma^2}}}{\sigma^2}$$

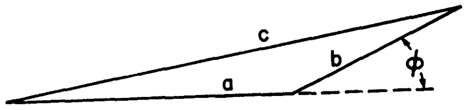
where R ≥ O. This probability density function Q_1(R) governs the distribution of the envelope and is known as the Rayleigh distribution. The parameter R is restricted to non-negative values. It should not be confused with the normal probability density function where the parameter may take on both positive and negative values. Solving the above integral $Q_2(\phi)$ probability density function for the phase angles ϕ(t) is given by $Q_2(\phi)$ = 1/2 π where 0 ≤ ϕ ≤ 2π. This shows that the values of ϕ(t) are uniformally distributed over zero to 2π and have a rectangular distribution.[3]

The Presence of a Parasitic Target Jittering In Phase in a Clutter Environment

Let a = No. 1 parasitic target signal, rms
b = No. 2 parasitic target signal, rms
c = the phasor sum of a and b
P_o = the mean square value of the random scatters
$m^2 = C^2/P_o = C^2$ for P_o normalized to $P_o = 1$.

Assume that the probability density function of ϕ is

$$f(\phi) = \frac{1}{\pi},\quad 0 \le \phi \le \pi$$



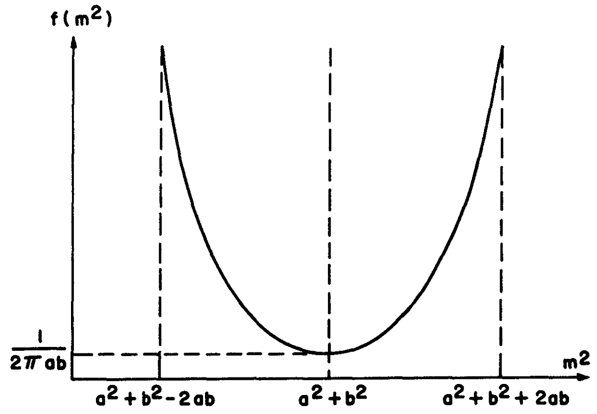
$$C^2 = a^2 + b^2 + \cos\phi = m^2$$

1. If y = 2ab cos ϕ, or $\phi = \cos^{-1}\left(\frac{y}{2ab}\right)$
since $f(y) = f(\phi)\ \left|\frac{d\phi}{dy}\right|$

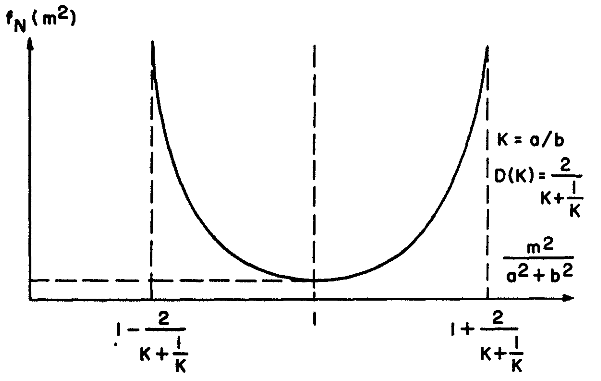
2. Therefore,

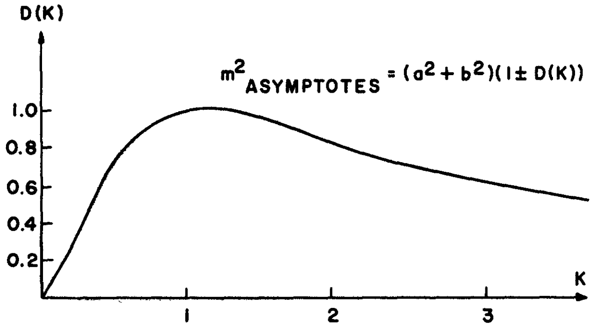
$$f(y) = \frac{1}{2\pi\ ab}\left[1 - \left(\frac{y}{2ab}\right)^2\right]^{-\frac{1}{2}}$$

3. or, the probability density function of m^2 is



This distribution indicates the probability, $f(m^2)d(m^2)$, of obtaining a particular type of first probability distribution in power for a target consisting of random scatters and two strong targets. It can be indicated as





As a typical case consider that the levels of the fixed targets have been established at a_1^2 and b_1^2, and that the parasitic-1 to parasitic-2 target ratio is $K_1 = a_1/b_1$, leaving only the relative phase between a_1 and b_1 unknown. Then, we can specify with what probability we can expect to obtain an m^2 (with corresponding type of first probability distribution)

$W_1(P/\bar{P})$, which may lie only in the range,

$$a_1^2 + b_1^2 - \frac{2}{K_1 + 1/K_1} \leq m^2 \leq a_1^2 + b_1^2 + \frac{2}{K_1 + 1/K_1} .$$

Note the following:

1. The value around which m^2 may range is $a_1^2 + b_1^2$.
2. The percentage spread in m^2 is dependent on $K = a/b$; i.e., $D(K) = \frac{2}{K + \frac{1}{K}}$
3. Although the probability of having an $m^2 \leq a_1^2 + b_1^2$ always remains at 1/2, the probability with which we can predict a given type of $W_1(P/\bar{P})$ increases with K, independent of $a^2 + b^2$. This implies that under the condition that K>10, the clutter generator provides stationary statistics.

On the basis of this analysis, it is recommended that the clutter environment be characterized by:

1. $a^2 + b^2$
2. $K = a/b$

in order to use the result that

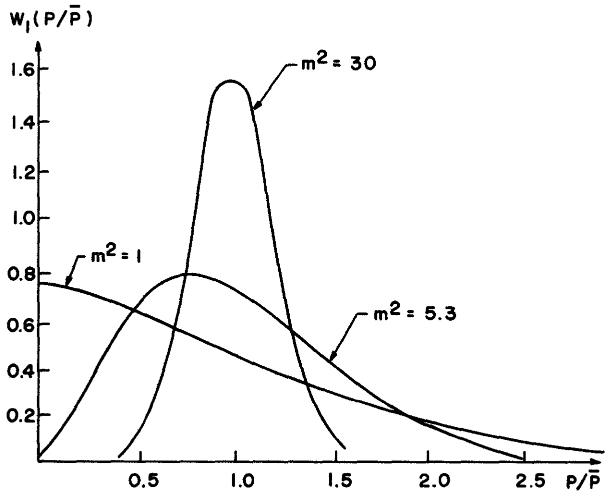
$$W_1(P)dP = (1 + m^2)\, e^{-m^2}\, e^{-P/\bar{P}\,(1+m^2)}\, J_0\left(2im\sqrt{1+m^2}\sqrt{\frac{P}{\bar{P}}}\right) \frac{dP}{\bar{P}}$$

First probability distribution in power for a target consisting of random scatters plus a fixed target, for several values of $m^2 = S^2/P_o$.

S^2 = steady power

P_o = random power

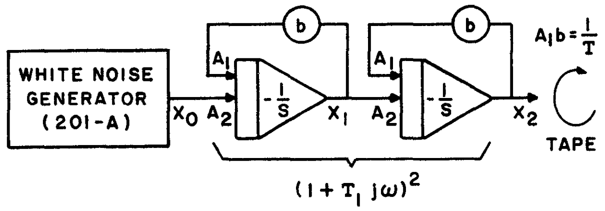
Ref: Vol. 13 Radiation Laboratory Series



The nature of $f(m^2)$ is such as to suggest that, in a large percentage of the possible $W_1(P/\bar{P})$ cases, identifying and resolving each of two strong targets in ground clutter is definitely possible. Since the most likely, and therefore most prominent, traces on an "A" scope will be those caused by returns that have been reinforced and those that have been cancelled, a comparison of the difference channel "A" scope presentation with the sum channel "A" scope presentation should enable an operator to identify and measure the range of each target.[4]

APPENDIX 2

Comparison Of The Tape Recording With The Filter Output

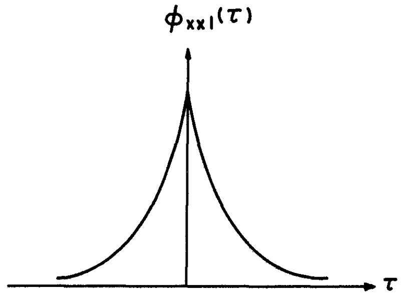


Let $\phi_{xx0}(\tau) = 2T_1\ \delta(\tau)$

1. If $H(j\omega) = (1 + jT_1\omega)^{-1}$

then $\phi_{xx1}(\tau) = e^{-|\tau|/T_1}$

where $G_{xx1}(\omega) = \dfrac{2T_1}{1 + T_1^2\omega^2}$



2. Hence

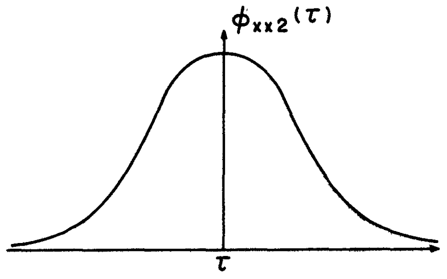
$$\phi_{xx2}(\tau) = \frac{1}{2T_1} \int_{-\infty}^{\infty} \phi_{xx1}(t_1)\ \phi_{xx1}(\tau - t_1)dt_1$$

3. Since the autocorrelation function is an even function, it will be necessary only to solve for the case of $\tau \geq 0$ and use the mirror image of the result for $\tau \leq 0$ in establishing the form of $\phi_{xx2}(\tau)$

$$\phi_{xx2}(\tau) = \frac{1}{2T_1}\left\{ \begin{array}{l} \int_{-\infty}^{0} e^{-\frac{2t_1 - \tau}{T_1}} u(-5_1)u(-t_1 + \tau)dt_1 \\ + \int_{0}^{\tau} e^{-\tau/T_1} u(t_1)u(-t_1 + \tau)dt_1 \\ + \int_{\tau}^{\infty} e^{-\frac{2t_1 + \tau}{T_1}} u(t_1)u(t_1 - \tau)dt_1 \end{array} \right\} \tau \geq 0$$

4. So,

$$\phi_{xx2}(\tau) = \frac{e^{-\tau/T_1}(|\tau| + T_1)}{2T_1}$$



5. It has been shown[5] that if we define

$$\sigma^2(\tau) = \sum \left\{ \left[\phi_{xx2}(\tau) - V(\tau) \right]^2 \right\}$$

where $V(\tau) = \frac{1}{T} \int_0^T x_2(t)x_2(t + \tau)dt$

and T = recording time,

then $\sigma^2(\tau) \leq \int_0^{\infty} [\phi(t)]^2\ dt$

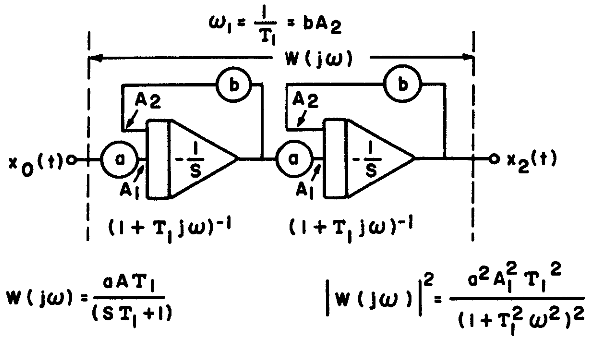
6. Substituting for $\phi(t)$,

$$\sigma^2 \leq \frac{4}{T} \int_0^{\infty} \left[\frac{e^{-t/T_1}(t + T_1)}{2T_1} \right]^2 dt$$

$$\therefore \sigma^2 \leq 5/4\left(\frac{T_1}{T}\right)$$

APPENDIX 3

Relationship Between The Noise Generator Output And The Filter Output



1. If

$$\Phi(j\omega) \triangleq \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{-j\omega\tau} \phi(\tau)d\tau$$

$$\phi(\tau) = \frac{1}{j} \int_{-j\infty}^{j\infty} e^{j\omega\tau} \Phi(j\omega)d(j\omega)$$

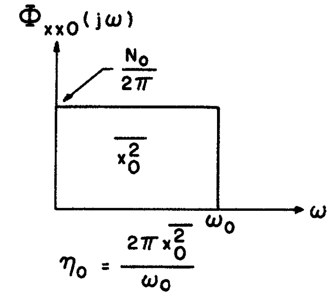
2. Then

$$\phi_{xx2}(0) = \int_{-\infty}^{\infty} \Phi_{xx2}(j\omega)d\omega$$

or

$$\overline{x_2^2} = \phi_{xx2}(0) = \int_{-\infty}^{\infty} \left|W(j\omega)\right|^2 \Phi_{xx0}(j\omega)d\omega$$

3. $$\overline{x_2^2} = \frac{a^2N_0A_1^2T^2}{2\pi} \int_{-0}^{\omega_0} \frac{d\omega}{(1+T_1^2\omega^2)^2}$$



4. $$\therefore \boxed{\overline{x_2^2} = \frac{a^2A^2}{\omega_1^2 2}\left[\frac{1}{1+(\omega_0/\omega_1)^2} + \frac{\omega_1}{\omega_0}\tan^{-1}\frac{\omega_0}{\omega_1}\right]\overline{x_0^2}}$$

$$\overline{x_2^2} = \frac{a_1^2A^2}{2}\left[\frac{1}{\omega_1^2+\omega_0^2} + \frac{1}{\omega_1\omega_0}\tan^{-1}\frac{\omega_0}{\omega_1}\right]\overline{x_0^2}$$

if $\omega_0 \ll \omega_1$

5. $$\boxed{\overline{x_2^2} \approx \frac{a_1^2A^2}{2}\left[\frac{1}{\omega_0^2} + \frac{1}{\omega_1\omega_0}\frac{\pi}{2}\right]\overline{x_0^2}}$$

REFERENCES

1. Kerr, Propagation of Short Radio Waves, Radiation Laboratory Series, Vol. 13, pp. 550-588, McGraw-Hill, 1951.

2. Lawson and Uhlenbeck, Threshold Signals, Radiation Laboratory Series, Vol. 24, McGraw-Hill, 1947.

3. Bendat, Principles and Applications of Random Noise Theory, John Wiley and Sons, 1958.

4. Shapiro, S.S. Hughes, Improvement of Angular Resolution by Means of Interpulse Frequency Modulation, AFCRC - TN - 59 - 754, on Contract No. AF 19(604)-2625, 15 June 1959.

5. Lanning and Battin, Random Processes in Automatic Control, McGraw-Hill.

THE SPECTRAL EVALUATION OF ITERATIVE DIFFERENTIAL ANALYZER INTEGRATION TECHNIQUES

M. C. Gilliland
Beckman/ Berkeley Division
Richmond, California

Summary

The iterative differential analyzer utilizes both continuous and incremental or iterative mathematical techniques for the solution of problems. Consequently, this machine is frequently required to perform numerical integration. In this connection it is desirable to evaluate numerical integration techniques from a control system point of view. The W-transform, a special case of the modified Z-transform, is used as a base for analysis.

The true numerical integration operator is shown to have the form $T(\log w)^{-1}$ where T is the time increment for the process. The truncation error for an integration process is related to the dynamic error of the corresponding integration operator which is determined by comparison with $T(\log w)^{-1}$. Various integration operators are evaluated with respect to dynamic error. Parasitic solutions are shown to correspond to poles of the W-transfer function of the operator. Round-off error accumulation may be determined from the W-plane representation of the operator. Synthesis of open-loop operators using root-locus and Bode techniques is discussed. Closed-loop operator selection and stability for the integration of linear ordinary differential equations is discussed. Techniques are indicated for the stabilization of closed-loop operators. Techniques are presented for the determination of truncation and round-off error estimates for the integration of linear ordinary differential equations.

Introduction

A new type of analog computer, the iterative differential analyzer (IDA), recently has been developed. This computer is more flexible than the electronic differential analyzer. The latter machine is capable of direct integration with respect to only one independent variable corresponding to machine time. The IDA, however, performs not only this operation, but with the addition of memory allows the programmer to use numerical techniques. The IDA can be used as a three or four significant figure digital computer performing integration numerically. The IDA can also be used as a hybrid computer where some computations are carried out numerically and others in a standard analog manner.

Since the IDA can use numerical integration techniques as part of its mathematical repertoire, it is helpful to compare these techniques from the control system point of view which is familiar to most analog computer users. It is desirable to be able to determine which numerical operator is the most suitable for a certain integration process, or further to compare the results of a numerical integration scheme with those to be obtained by direct integration.

W-Transform

Numerical integration can be thought of as a sampled-data process. One can use a transform algebra to analyze a sampled-data system in a manner similar to the use of the Laplace transform for the analysis of a continuous-data system. The modified Z-transform[1] has enjoyed wide application for this purpose. For the discussion which follows, it is convenient to use this transform with a delay which is equal to the sampling period. The resulting special case of the modified Z-transform is called the W-transform.[2] The relationship of the W-transform to the Z-transform is similar to that of the Laplace transform to the operational transform used by Heaviside. Some of the pertinent properties of the W-transform are summarized below.

W-Transform and Inverse

The W-transform of a continuous function, f(t), is defined by

$$W_T\{f(t)\} \triangleq \sum_{n=0}^{\infty} f(nT)w^{-(n+1)} = F_T(w).$$

The subscript, T, is omitted where the dependence of the transform on T is understood. Consideration of the well-known sampling theorem[3] leads to the conclusion that if f(t) is continuous and its Fourier spectrum vanishes identically for $\omega \geqslant \frac{\pi}{T}$, then f(t) is uniquely determined from a knowledge of F(w). Under these conditions, f(t) can be found from[2]

$$f(t) = \sum \text{Residues}\,[\,w^{\frac{t}{T}} F(w)\,] \qquad (1)$$

where the implied integration contour, encloses all the poles of F(w).

Region of Stability[1, 2]

If all the poles of F(w) lie inside the unit circle, $|w| < 1$, in the complex w-plane, then F(w) is the W-transform of a "stable" function, f(nt). That is, ultimately, $f(nt) \to 0$ with increasing n. If some poles of F(w) lie in $|w| > 1$, or if poles of order >1 lie on $|w| = 1$, then the corresponding f(nt) is an "unstable function". Ultimately, $|f(nt)| \to \infty$ with increasing n. If F(w) has no poles in $|w| > 1$ and only first-order poles on $|w| = 1$ then the corresponding f(nt) is "marginally stable". f(nt) does not tend to zero but is bounded for all n. This behavior is illustrated by the three functions and their W-transforms below.

f(t)	F(w)
$(0.1)^t$	$\dfrac{1}{w - 0.1^T}$
$\sin\omega t$	$\dfrac{\sin\omega T}{(w-\cos\omega T)^2 + \sin^2\omega T}$
2^t	$\dfrac{1}{w - 2^T}$

Steady-state Response of W-Transform Operators

The W-transform of $\sin\omega t$ is given by

$$F(w) = \frac{\sin\omega T}{(w-\cos\omega T)^2 + \sin^2\omega T}.$$

If we take $\theta = \omega T$, then

$$F(w) = \frac{\text{Sin}\theta}{(w-\cos\theta)^2 + \text{Sin}^2\theta}$$

and the poles of F(w) are seen to be located at $w = e^{\pm j\theta}$. Suppose O(w) is an operator on F(w) which results in the response H(w), so that $H(w)/F(w) = O(w)$. Then, the steady-state response $H(e^{j\theta})$ due to the excitation $F(e^{j\theta})$ is given by $O(e^{j\theta})F(e^{j\theta})$. If O is expressed in the polar form $O(e^{j\theta}) = G(\theta)e^{j\phi(\theta)}$, then G, ϕ are respectively the transfer gain and phase of the operator, O, as a function of θ, which may be called the "sample angle". Bode plots can be constructed for W-transfer functions from a knowledge of G, ϕ. It is seen that the sample limit given by the sampling theorem[3] corresponds to the sample angle $\theta = \pi$.

W-Transform of Finite Difference Operators

It can be shown that

$$W\{f(nT + mT)\} = w^m\, W\{f(nT)\} = w^m F(w).$$

This result can be applied to the finite difference operators

$$Ef(nT) \equiv f(nT + T)$$

$$\Delta f(nT) \equiv f(nT + T) - f(nT)$$

$$\nabla f(nT) \equiv f(nT) - f(nT - T)$$

to give

$$W\{Ef(nT)\} = wF(w)$$

$$W\{\Delta f(nT)\} = (w-1)F(w)$$

$$W\{\nabla f(nT)\} = \left(\frac{w-1}{w}\right)F(w),$$

which define the W-transfer functions for these operators. W-transfer functions can be deduced for more complex finite difference operators in an obvious way.

The True Numerical Integration Operator

Suppose f(nT) is some sampled-data function. One might ask if there is a unique continuous function, g(t), such that g'(nT)=f(nT). If so, then apparently g(nT) can be considered the "true integral" of f(nT). This question is basic to the problem of numerical integration. Assuming that g(nT) exists, if an operator, O, can be found such that

$$G(w)=O(w)\,F(w)$$

for some set of functions $f_\alpha(nT)$, then O is a "true numerical integration operator" with respect to these f_α. g(nT) does exist and an O can be found whenever the f_α satisfy certain properties.

If f(nT) is not singular for finite n, then f has a unique continuous representation h(t) with the restriction that the Fourier spectrum for h vanish identically for $\omega \geqslant \frac{\pi}{T}$. Thus,

h(nT)=f(nT). h is a valid representation for f, since for practical cases T must be restricted so that no significant information is lost. Then, the true integral of f(nT) is g(nT) where $g(t)=\int h(t)dt$. Suppose g(t) is given by (see equation (1) above)

$$g(t)=\frac{1}{2\pi j}\int_C w^{\frac{t}{T}}G(w)dw$$

where C is suitably chosen and G(w) is the W-transform of g(t). It can be shown that

$$g'(t)=\frac{1}{2\pi j}\int_C w^{\frac{t}{T}}\,\frac{\log w}{T}\,G(w)dw,$$

and

$$W\{g'(t)\}=\frac{\log w}{T}G(w).$$

It follows that

$$G(w)=\frac{T}{\log w}F(w)$$

and O(w), the true numerical integration operator, is given by

$$O(w)=\frac{T}{\log w}.$$

The steady-state transfer function for O is

$$O(e^{j\omega T})=\frac{1}{j\omega}$$

as one might suspect. This result can be developed on an alternative and intuitive basis. Suppose h(t), the continuous representation of f(nT), can be considered a linear combination of functions

$$\sum_{n=0}^{\infty} A_n e^{j(n\bar{\omega}t-\phi_n)}$$

In this event, g(t) is given by

$$\sum_{n=0}^{\infty}\frac{A_n}{j\omega n}e^{j(n\bar{\omega}t-\phi n)}.$$

For each term

$$W\{\frac{A_n}{j\bar{\omega}n}e^{j(n\bar{\omega}t-\phi n)}\}=\frac{1}{j\bar{\omega}n}W\{A_n e^{j(n\bar{\omega}t-\phi n)}\}$$

Thus in the limit, the above result obtains.

Unfortunately $\frac{T}{\log w}$ has no realizable representation in terms of numerical processes. However, it can be used to provide a spectral estimate of the integration error for a given integration operator.

Dynamic Error

In order to evaluate the useful range of the sample angle, θ, for a numerical integration operator, its phase and gain as a function of θ can be compared with that for $T(\log w)^{-1}$. Thus, a Bode analysis can be made of the spectral integration properties of an arbitrary integration operator.

Dynamic error is generally a more meaningful measure of the integration capability of an operator. Suppose g(nT) is the true integral of some function f(nT). If an operator, O, acting on f produces the response $\bar{g}(nT)$, then what is really desired as a measure of integration error is an estimate of $E=\mathrm{Max}\{|g-\bar{g}|/|g|\}$. E can be computed as a function of θ (or ω if T is fixed). If $\bar{G}(w)=O(w)F(w)$, then the steady-state transfer function for O can be represented

$$O(e^{j\theta})=G(\theta)\,e^{j\phi(\theta)}.$$

The steady-state form of the true integration operator is

$$\frac{1}{\omega} e^{-j\pi/2}$$

so that

$$\frac{|g-\bar{g}|}{|g|} = \left| 1 - \frac{\bar{g}}{g} \right| = \left| 1 - \omega G(\theta) e^{j(\phi + \frac{\pi}{2})} \right|.$$

Thus

$$E_\theta = 1 + (\omega G)^2 + 2(\omega G) \sin\phi.$$

In what follows it will be seen that the dynamic error of an integration operator is the same as its normalized spectral truncation error.

Truncation Error

For a discussion of truncation error, it will be convenient to denote f(nT) by f_n. Thus f_{n+m} means f(nT+mT). If

$$y(t) = \int f(t)dt$$

then

$$y_{n+1} = y_n + \int_{t=nT}^{t=(n+1)T} f(t)dt.$$

In order to determine y_{n+1} precisely, f(t) must be known for almost all t in the interval $nT < t < (n+1)T$. However, for numerical integration, f(t) is generally known only for discrete values of t, resulting in the sampled-data function f(mT). Thus, some approximation must be found for

$$H(t) = \int_{t=nT}^{t=(n+1)T} f(t)dt$$

in terms of f(mT), which will provide an adequately accurate estimate of y_{n+1}. H(t) is usually approximated by the application of some finite difference operator to f(nT). An operator, O, is said to be closed or open depending on whether Of(nT) involves a knowledge of f(mT) for $m \leqslant n+1$ or $m \leqslant n$.

Open Integration Operators

An open integration operator has the form

$$O\,f(nT) = \left[\sum_{j=1}^{k} TA_j\, \nabla^j \right] f(nT), \qquad (2)$$

where k and the A_a are suitably chosen so that Of(nT) approximates $y_{n+1} - y_{n-p}$. There are numerous special cases[4] of equation (2) which are commonly used either for predictors or for integration. Three are given below as an example.

$$y_{n+1} = y_n + T f_n \quad \text{(Euler)}$$

$$y_{n+1} = y_n + \frac{T}{24}(55f_n - 59f_{n-1} + 37f_{n-2} - 9f_{n-3}) \text{(Adams)}$$

$$y_{n+1} = y_{n-3} + \frac{4T}{3}(2f_n - f_{n-1} + 2f_{n-2}) \text{(Newton-Cotes)}$$

The W-plane representation of equation (2) is

$$\sum_{j=1}^{k} TA_j \left(\frac{w-1}{w}\right)^j$$

so that the W-transfer function of an open operator from y_{n-p} to y_{n+1} is

$$\lambda_o(w) = \left(\frac{w^p}{w^{p+1}-1}\right)\left[\sum_{j=1}^{k} TA_j \left(\frac{w-1}{w}\right)^j\right]. \qquad (3)$$

Closed Integration Operators

A closed integration operator has the form

$$Of(nT) = \left(\sum_{j=1}^{k} TA_j\, \nabla^j\right) f(nT+T), \qquad (4)$$

where k and the A_a are again suitably chosen so that Of(nT) approximates $y_{n+1} - y_{n-p}$.

Three examples of special cases are given below.

$$y_{n+1} = y_n + \frac{T}{2}(f_{n+1} + f_n) \quad \text{(Trapezoidal)}$$

$$y_{n+1} = y_{n-1} + \frac{T}{3}(f_{n+1} + 4f_n + f_{n-1}) \quad \text{(Simpson)}$$

$$y_{n+1} = y_{n-2} + \frac{3T}{8}(f_{n+1} + 3f_n + 3f_{n-1} + f_{n-2}) \quad \text{(Newton)}$$

The W-plane representation of (4) is

$$w\sum_{j=1}^{k} TA_j \left(\frac{w-1}{w}\right)^j$$

so that the W-transfer function of a closed operator from y_{n-p} to y_{n+1} is

$$\lambda_c(w) = \left(\frac{w^{p-1}}{w^{p+1}-1}\right)\left[\sum_{j=1}^{k} TA_j \left(\frac{w-1}{w}\right)^j\right]. \qquad (5)$$

Truncation and Dynamic Error

The error committed by representing $T(\log w)^{-1}$ with an expansion of finite order in the operator ∇ is called the truncation error. The coefficients A_a in equations (2), (4) are usually determined so that f(nT) is approximated by a polynomial in t of degree k+2. Since only a limited class of functions are well approximated by polynomials, there is no guarantee that the A_a are optimal for a general purpose integration scheme. Further, the error estimates based on this procedure in many cases represent only very conservative upper bounds. Since every well-behaved sampled-data function, f(nT), has a continuous representation, h(t), which has a finite spectrum, it is preferable to evaluate truncation error spectrally. The total error in the computation of y_{n+1} from a knowledge of $f(mT), m \leqslant n+1$ and y_{n-p} is equal to the inner product of the dynamic error and the spectral representation of f(nT). Consequently, the dynamic error is a spectral representation of the truncation error. Fig. 1 shows the dynamic error of various integration operators as a function of the sample angle, θ.

Note that one five-point closed operator has better error properties than the five-point Newton-Cotes operator. This shows that the A_a are not chosen in an optimal way in the latter formula. The round-off error stability (to be discussed later) of both are comparable.

Parasitic Roots for Open-Loop Integration

In the process of constructing a numerical integration operator (see equations (3), (5)) which approximates $T(\log w)^{-1}$, parasitic roots are introduced which correspond to poles of the operator in the w-plane for an open-loop system (see Fig. 2). These roots are excited by round-off error, other noise, or the normal excitation of the system and generate spurious parasitic responses. The open-loop stability of an operator can be determined from the location of these poles in the w-plane. Note that the parasitic roots for a closed-loop system do not in general correspond to the poles of the operator. The closed-loop behavior of integration systems will be discussed later.

The stability properties of an open-loop operator are illustrated with examples. The transfer function for Simpson's rule is given by

$$\frac{T}{3}\cdot\frac{w^2+4w+1}{w^2-1}\,.$$

The poles of this operator are located at $w=+1, -1$. The pole at $w=+1$ is a basic requirement for an integrator in the same sense that a pole is needed at $s=0$, in the s-plane for integration in a continuous-data system. The pole at $w=-1$ results in a marginally stable parasitic response of the Simpson operator. Note that this root causes a resonance of the operator to uniformly alternating round-off error, e, whose W-transform is (for an ϵ magnitude)

$$E(w) = \frac{\epsilon}{(w+1)^2}\,.$$

The transfer function for Newton's rule is given by

$$\frac{3T}{8}\cdot\frac{(w+1)^3}{w^3-1}\,.$$

The parasitic roots of this operator correspond to poles located at $w=e^{j\frac{2\pi}{3}}, e^{j\frac{4\pi}{3}}$.

It is seen that the parasitic response is again marginally stable.

Round-Off Error Sensitivity

Frequently, the round-off error distribution is known for an integration process. In this case, a round-off error spectrum can be found which then can be used to determine quantitatively the response of a stable system to round-off error. In any event, a realistic upper bound for round-off error accumulation can be obtained from the knowledge of the location of the roots of a stable system.

Suppose a system root is located at $w=ae^{j\theta}$, $a<1$. Further, suppose the system is excited sinusoidally by a driving function whose poles are located at $w=e^{\pm j\phi}$. Then the response of the system corresponding to the root at $w=e^{j\theta}$ is magnitude bounded by

$$[1+a^2-2a\cos(\theta-\phi)]^{-0.5}.$$

Maximum response is seen to obtain for $\theta=\phi$, and is given by $(1-a)^{-1}$. Thus, one can assume (and conservatively so) that all the round-off error as a function of nT has the form $\epsilon\cos(n\theta)$ to obtain an upper bound for this particular system response to round-off. The above procedure can be used to obtain a bound for each system response. Then a bound for the overall effect of round-off error on the system is a weighted average of these bounds.

If the system has unstable roots then, ultimately, the response of the system due to round-off error alone will be dominant. The long-time response due to an unstable system root is essentially independent of excitation whose growth is no more than exponential. Consequently, a good estimate can be made of the contribution of unstable roots to total system response. In this connection it is frequently important to be able to establish an upper bound on the number of computation cycles undergone by a slightly unstable operator before the error growth is deleterious. The root-locus chart for the system provides this capability.

Closed-Loop System Stability

Integration operators are used more often to integrate, numerically, a simultaneous set of ordinary differential equations than to perform an open-loop integration of some function. The parasitic roots of these operators generally determine the stability characteristics of the integration system. In order to evaluate the stability properties of a closed-loop system it is necessary to determine the location of the closed-loop roots. This can be accomplished from a knowledge of the open-loop poles and zeros for a linear system. Root-locus techniques are advantageous for this purpose. Frequently, much can be determined about the stability characteristics of a non-linear system using linear analysis.

It will suffice to use two simple differential equations to illustrate the techniques suggested. These are $(y=y(x))$

$$y' = \alpha y + f(x) \qquad (6)$$

$$y' = -\alpha y + f(x) \qquad (7)$$

whose solutions are respectively unstable and stable.

Stable Differential Equations

Equation (7) is a simple case of a stable differential equation. It can be written in the form

$$y = -\int \alpha y\,dx + \int f dx.$$

Taking the W-transform with respect to x we have

$$Y = -\frac{\alpha T}{\log w} Y + \frac{T}{\log w} F.$$

Suppose $T(\log w)^{-1}$ is approximated by G(w). Then

$$Y = G(w)\,[F - \alpha y].$$

A block diagram of this system is shown in Fig. 3. The open-loop poles and zeros for the system are those of G. The closed-loop roots satisfy $\alpha G = -1$. G contains the parameter T and consequently G can be expressed

$$G = T\bar{G}.$$

Thus, the closed-loop roots can be determined as a function of $K=\alpha T$ from a root-locus plot for $K\bar{G} = -1$. System stability depends on whether the roots are inside or outside the unit circle, $|w|=1$, in the w-plane.

The stability plots as a function of K for various operators used in the integration system of Fig. 3 are shown in Figs. 4 - 13 (solid lines). The operators are given below.

Operator	Operator and Figure Number
$\frac{T}{w-1}$	4 (Euler)
$\frac{T}{2} \cdot \frac{w+1}{w-1}$	5 (Trapezoidal)
$\frac{T}{3} \cdot \frac{w^2+4w+1}{w^2-1}$	6 (Simpson)
$\frac{3T}{8} \cdot \frac{(w+1)^3}{w^3-1}$	7 (Newton)
$\frac{2T}{45} \cdot \frac{7w^4+32w^3+12w^2+32w+7}{w^4-1}$	8 (5-point Newton Cotes)

$$\frac{T}{3.27} \cdot \frac{w^4+5.8w^3+6w^2+5.8w+1}{w^4+w^3-w-1} \qquad 9$$

$$\frac{T}{24} \cdot \frac{9w^3+19w^2-5w+1}{w^3-w^2} \qquad 10$$

$$2T \cdot \frac{w^2+2w}{5w^2-4w-1} \qquad 11$$

$$\frac{T}{4} \cdot \frac{w^3+11w^2+11w+1}{w^3+3w^2-3w-1} \qquad 12$$

$$\frac{T}{5} \cdot \frac{w^4+26w^3+66w^2+26w+1}{w^4+10w^3-10w-1} \qquad 13$$

It can be seen that operators 13, 12 are seriously unstable and are of limited use for either open- or closed-loop systems. For very small K, one root of operator 13 represents an unstable system response of $(-10)^n$, where n is the computation step number.

Operators 6, 7, 8, 9 can be used for short-time integration with small K. Of these operator 7 has the best stability characteristics. Operators 4, 10 can be used for long-time integration with limited open-loop gain (limited Δx). Operators 5, 11 can be used for long-time integration with any finite non-zero open-loop gain.

It must be recognized that stability is not the only requirement which limits the open-loop gain of the system. The gain is a function of the sample angle and one must also limit the sample angle so that the dynamic error of the operator is not serious. The dynamic error curves for the above operators are shown in Fig. 1.

Root-locus stability plots may be made in a similar manner for more complex linear integration systems.

Unstable Differential Equations

Equation (6) is simple case of an unstable differential equation. It can be written in the form

$$y=\int \alpha y dx + \int f dx$$

which leads to the transformed equation

$$Y=G(w)[F+\alpha y]$$

where G(w) is an approximation of $T(\log w)^{-1}$. The block diagram for this system is the same as that shown in Fig. 3 except for the feedback, which in this case is positive. The closed loop roots satisfy $K\bar{G}=1$ where $K=\alpha T$, $G=T\bar{G}$.

The unstable system stability plots for operators 14-7 are shown in Figs. 14-7 with hatched loci. Note that no integration operator whose dynamic error is zero for zero sample angle can result in stable system response. An evaluation can be made of the usefulness of a particular operator by a comparison of the growth of the unstable system response due to the operator with the growth of the desired unstable solution of the differential equation.

Stability Compensation

Sometimes an integration system can be stability compensated by the addition of suitably located poles and zeros in the w-plane. Generally, this technique provides a less rapid growth of the unstable response rather than the elimination of unstable roots. Compensation has an effect on the dynamic error of the operator which also must be considered.

Synthesis of Integration Operators

Generally, there are four criteria for the synthesis or selection of an integration operator. These are:

1) Its influence on the stability characteristics of the closed-loop system in which it is to be used.

2) Whether the integration is long- or short-time.

3) Whether it has suitable dynamic error.

4) How much machine time is required for its use.

These criteria cannot all be satisfied independently. However, the use of control system synthesis techniques simplifies the development of an operator which has desirable characteristics with respect to these criteria.

Series Approximation of $T(\log w)^{-1}$

Although the following method for operator synthesis is not particularly fruitful, it is described because of its interesting relationship to well-known integration formulae. A common series expansion for log w is

$$\log w = 2\left[\frac{w-1}{w+1} + \frac{1}{3}\left(\frac{w-1}{w+1}\right)^3 + \frac{1}{5}\left(\frac{w-1}{w+1}\right)^5 + \cdots\right], |w| > o$$

If various numbers of terms are retained to approximate log w and the result is used to generate an approximation for $T(\log w)^{-1}$, a sequence of integration operators will result. For example, if one term of the series is retained then

$$T(\log w)^{-1} \sim \frac{T}{2} \cdot \frac{w+1}{w-1}$$

which is the trapezoidal operator. It two terms are retained,

$$T(\log w)^{-1} \sim \frac{3T}{8} \cdot \frac{(w+1)^3}{w^3-1}$$

which is the Newton operator.

Polynomial Operators

One might intuitively suspect that an operator selected for its ability to integrate polynomials should have improved dynamic error properties. In order to evaluate such a scheme, some of these operators are developed below and then evaluated with respect to stability and dynamic error. Such a sequence of polynomial integration operators is defined by

$$O_n(w) = \frac{w\{t^n/n!\}}{w\{t^{n-1}/(n-1)!\}}, n > o.$$

F our of these are

$$O_1(w) = \frac{T}{w-1} \qquad \text{(Euler)}$$

$$O_2(w) = \frac{T}{2} \cdot \frac{w+1}{w-1} \qquad \text{(Trapezoidal)}$$

$$O_4(w) = \frac{T}{4} \cdot \frac{w^3+11w^2+11w+1}{w^3+3w^2-3w-1}$$

$$O_5(w) = \frac{T}{5} \cdot \frac{w^4+26w^3+66w^2+26w+1}{w^4+10w^3-10w-1}.$$

O_1, O_2 are well-known. O_4, O_5 are of limited use because they are both closed and highly unstable (see Figs. 13, 12). These are operators 11, 13 above. However, it is interesting to note that the dynamic error of the latter two is quite small.

Optimization of Coefficients

It was noted earlier that the coefficients usually used in the expansions of equations (3), (5) are not optimal. Frequently, a readjustment of the poles of zeros of the operator will lead to smaller dynamic error without significantly changing its stability characteristics. For example operator 9 (not O_9) is five-point and closed. It is a linear combination of the Newton and Simpson operators. It has approximately the same dominant stability characteristics as the Simpson operator. Compared with any other standard five-point closed operator, it exhibits less dynamic error. It has about the same general stability properties as the five-point Newton-Cotes operator. Obviously, the coefficients used for operator 9 are a better choice than those for other standard five-point formulae.

As an example of how the coefficients of an operator may be adjusted for stability at the expense of dynamic error, compare operator 11 with the Simpson operator. Both are three-point and closed. Dynamic error is smaller for the Simpson operator but operator 11 has better stability properties.

The IDA can be used as a tool for the study of integration operators. It is capable of generating root-locus plots and determining dynamic error. One may conveniently use the high speed iterative feature to optimize the dynamic error properties of an operator empirically.

Choice of Operator for a Closed-Loop System

One technique for the choice of an operator to be used for the numerical integration of some particular system of ordinary differential equations is outlined below. What follows assumes that the programmer has some gross knowledge of the characteristics of the solution.

Linear Ordinary Differential Equations

The discussion here will be restricted to those equations of the constant coefficient type.

Usually, the extension of the technique to a more general linear case is not unduly difficult.

First, the use of an integration operator must result in suitable closed-loop system stability properties. This includes consideration of total solution time, type of solution expected, etc. The stability restriction will eliminate from consideration many of the integration operators in the programmer's repertoire.

Second, some estimate must be made of the bandwidth of the system response, and what dynamic error is tolerable within this bandwidth. This will fix T (the integration increment) for any particular operator. Some operator can then be chosen based on simulation time limits. The choice should be rechecked with respect to stability.

It should be noted that this is only one approach. Others exist and are often more desirable, using results developed earlier. For example, it may be more practical to evaluate the effect of truncation error on the basis of closed-loop system response rather than on an open-loop dynamic error basis.

Non-Linear Differential Equations

The techniques outlined for linear systems frequently can be used to obtain bounds for round-off error accumulation and truncation error estimates. If gross information is available about magnitude bounds for dependent variables as well as expected bandwidth requirements, then sometimes a "linearized worst case" approach can be taken. For example, the solution of many non-linear equations can be approximated by a sequence of linear solutions. With this technique, stability requirements are established by the properties of the most severe linear solution. In a similar manner, the necessary dynamic error level for the integration operator can be estimated.

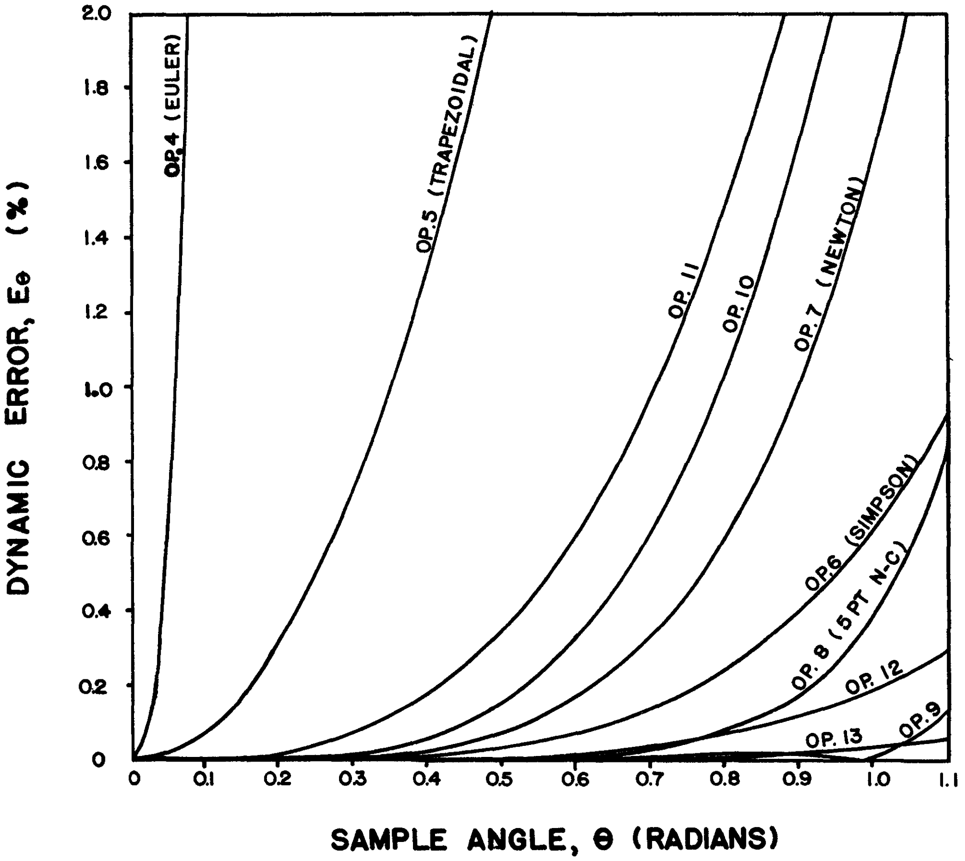
References

1. Jury, E.I., Sampled-Data Control Systems, Wiley, 1958.

2. Gilliland, M.C. The W-Transform, Presented at the Northwest Joint Computer Conference, October 1, 1960.

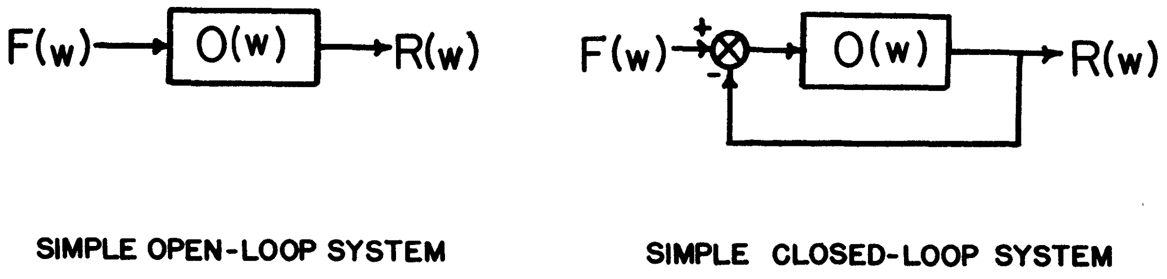
3. Shannon, C.E., Proceedings of the I.R.E., Vol. 37, January, 1949, pp. 10-21.

4. Hildebrand, F.B., Introduction to Numerical Analysis, McGraw-Hill, 1956.



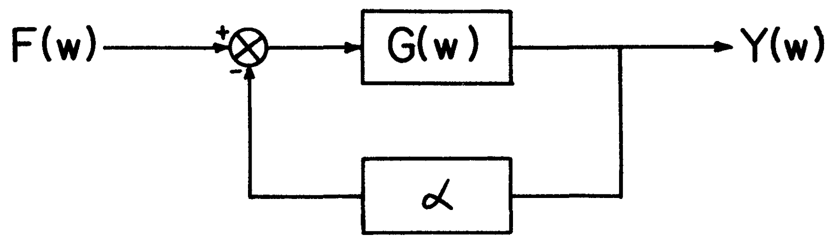
DYNAMIC ERROR OF VARIOUS INTEGRATION OPERATORS

FIGURE 1



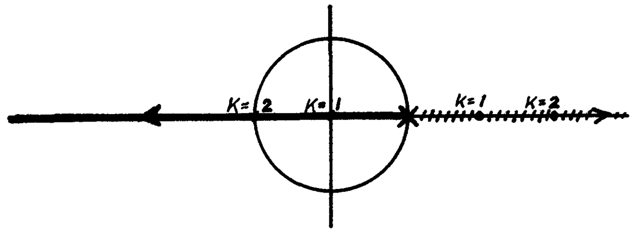
SIMPLE OPEN- AND CLOSED-LOOP INTEGRATION SYSTEMS

FIGURE 2



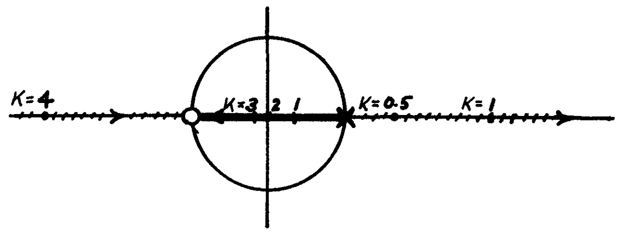
INTEGRATION SYSTEM FOR $y' = -\alpha y + f$

FIGURE 3



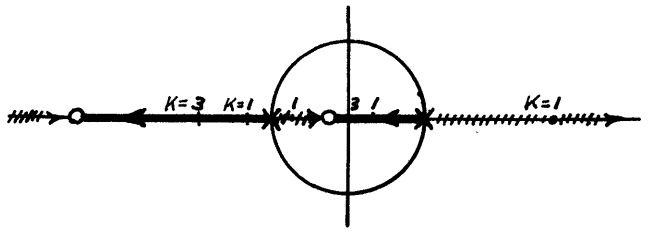
$O(w) = \frac{T}{w-1}$ (EULER)

FIGURE 4



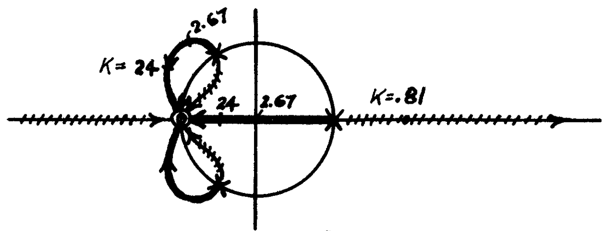
$O(w) = \frac{T}{2} \cdot \frac{w+1}{w-1}$ (TRAPEZOIDAL)

FIGURE 5



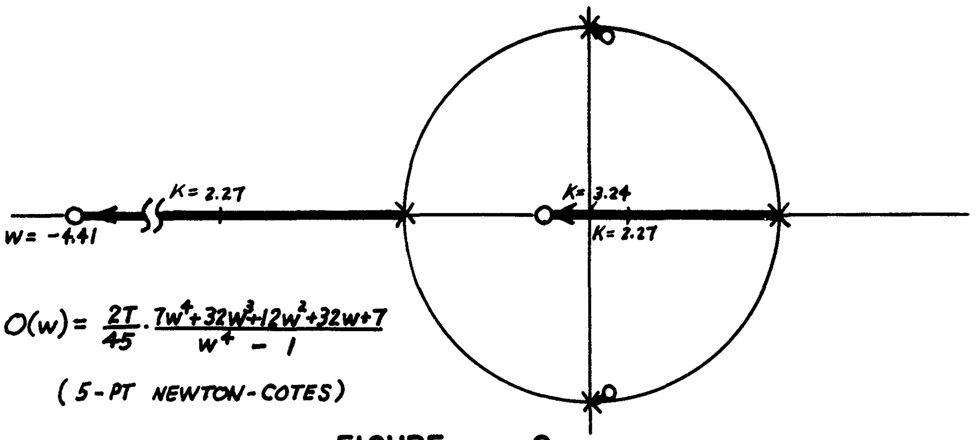
$O(w) = \frac{T}{3} \cdot \frac{w^2+4w+1}{w^2-1}$ (SIMPSON)

FIGURE 6



$O(w) = \frac{3T}{8} \frac{(w+1)^3}{w^3-1}$ (NEWTON)

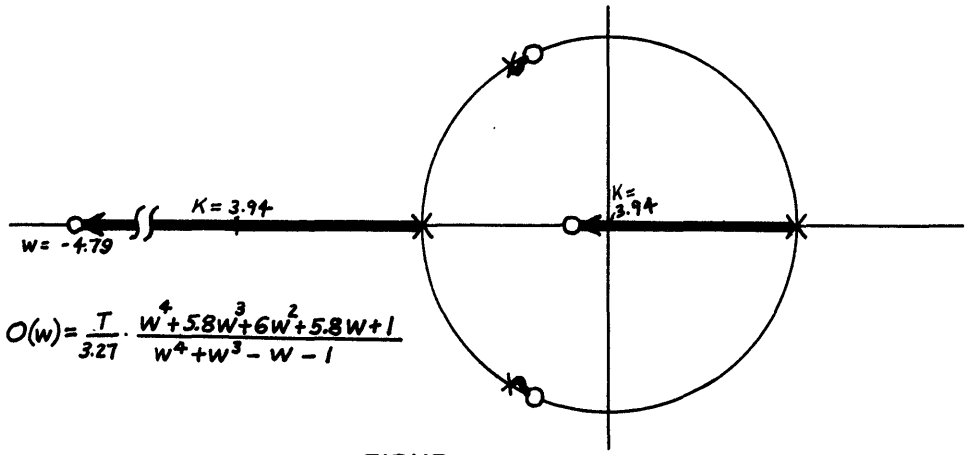
FIGURE 7



$$O(w)=\frac{2T}{45}\cdot\frac{7w^4+32w^3+12w^2+32w+7}{w^4-1}$$

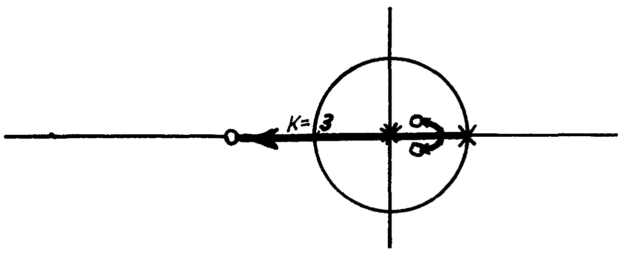
(5-PT NEWTON-COTES)

FIGURE 8



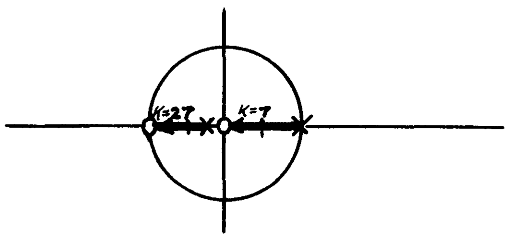
$$O(w)=\frac{T}{3.27}\cdot\frac{w^4+5.8w^3+6w^2+5.8w+1}{w^4+w^3-w-1}$$

FIGURE 9



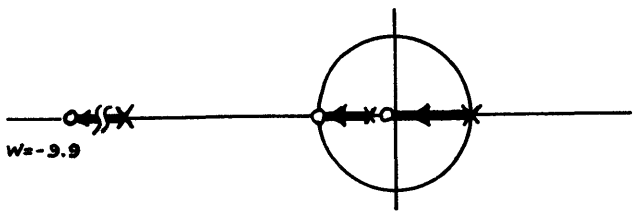
$$O(w)=\frac{T}{24}\cdot\frac{9w^3+19w^2-5w+1}{w^3-w^2}$$

FIGURE 10



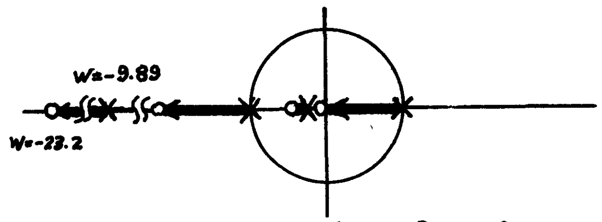
$$O(w)=2T\cdot\frac{w^2+2w}{5w^2-4w-1}$$

FIGURE 11



$$O(w)=\frac{T}{4}\cdot\frac{w^3+11w^2+11w+1}{w^3+3w^2-3w-1}$$

FIGURE 12



$$O(w)=\frac{T}{5}\cdot\frac{w^4+26w^3+66w^2+26w+1}{w^4+10w^3-10w-1}$$

FIGURE 13

AN ITERATION PROCEDURE FOR PARAMETRIC MODEL BUILDING AND BOUNDARY VALUE PROBLEMS

Walter Brunner
Electronic Associates, Incorporated
Princeton, New Jersey

Summary

A steepest descent iterative approach based on least squares error minimization is employed to obtain the optimum set of parameters typifying a system such that the system outputs satisfy desired performance criteria. The iteration process is shown to converge under fairly general conditions provided the initial guess lies in a neighborhood of the true optimum. An automatic gain control assures stability of the iteration procedure, which is a single step iteration process, i.e., only one parameter is updated at a time, which aside from theoretical considerations minimizes equipment duplication. The optimization procedure may also be applied to the problem of determining the parameters defining a physical system by comparing the outputs of the physical system to those of an approximate model simulated on an analog computer. The errors due to the comparison are minimized by the optimization procedure to determine the parameters of interest. The same procedure can be applied to home in on boundary conditions in ordinary differential equations.

Introduction

One of the problems frequently encountered in simulating a physical system on an analog computer is the problem of finding the values of the parameters typifying the system so that the system outputs will have desired performance characteristics. Optimizations of this type can involve many computer runs even when a good understanding of the physical system exists. Since if the number of independent parameters is say, N, then one must search for that point in N - dimensional space yielding the desired output characteristics. The magnitude of this task is clear. The object of this paper is to present an approach which automates this optimization process so that the system engineer can simply specify the performance criteria and then let the computer itself do the laborious task of finding the optimum set of parameters. The optimization process under consideration is pictured in Fig. (1). The system parameters α can be considered to be inputs to the system in the sense that they are varied until the system outputs, y(t), behave in the desired fashion as determined by the performance criteria. The outputs $\mathcal{E}(t)$, of the box labeled performance criteria are error quantities and are a measure of how well the individual performance criteria are satisfied. Since all these errors have to be simultaneously evaluated, a single measure of total system performance, S, has been chosen which in this case is a weighted integral least squares error criterion.*

To date, the design engineer optimizing a physical system on an analog computer will, in general, adjust the parameters, α, in a trial and error fashion by observing the system outputs, y(t), until the desired response is obtained. The design engineer is acting as the feedback element by mentally evaluating the system characteristics and employing his judgement and experience to readjust the parameters, α. If he is more sophisticated, he will have instrumented on the computer the performance criteria and a total measure of system performance, S, as in Fig. (1); if in addition a high speed repetitive computer is at his disposal his task as feedback element in trying to determine the optimum set of parameters, α, is considerably eased since he can observe whether S is being minimized on the display unit as he is adjusting the parameters. However, if the problem is nonlinear, or the number of parameters is large, then automatic techniques are a necessity. Meissinger[1] by means of the parameter influence coefficient technique has been able to obtain the gradient of S with respect to the parameters, α, on an analog computer - thus indicating the direction of the vector in which the parameters should be adjusted. In order to make the process completely automatic, it is necessary to know not only the direction of the vector correction, but also the length of the vector required such that the iteration process converges. An iteration process is required because it is desired to obtain the "best single value" of the parameters for the entire time history under consideration.

Margolis and Leondes[2] employ a similar approach, however, the problem which they are solving is not the one described here. They are interested in determining the values of the parameters, α, which minimize S at each instant of time, t. The answers which they obtain are then functions of time - that is each α is a time function. Therefore, their solution does not require an iteration process.

*One could have chosen other measures of total system performance such as

$$S=\sum_{i=1}^{I}\int_{0}^{T}G_i(t)|\mathcal{E}_i(t)|dt, \text{ etc.}$$

However, the technique presented applies only to least squares error criteria.

Shapiro[3] developed a digital technique which is mathematically equivalent to the one presented in this paper, where the differential equations of the system are expanded into a Taylor time series, and the errors are minimized by a least squares criterion. The feasibility of automatic solution by analog methods has been made possible by the advent of the parameter influence coefficient technique[1] which obviates the necessity of the Taylor time series expansion. The chief advantage of the analog computer being in obtaining solutions quickly, and hence the feasibility of solving "on line" problems.

A real time computer is perfectly adequate for most applications because relatively few iterations are needed to converge and small solution times are not essential. Sample-hold operations, and automatic cycling of the computer modes (RESET, HOLD, OPERATE) for the iteration process is accomplished by externally controlling the hold and reset relays. Of course, if a high speed repetitive computer is available the solution time will be negligible. In any case the real time mode is more convenient for check out purposes and final recording of results.

Practical examples where the optimization procedure was applied are the following: 1) Fitting the response of a complex nonlinear three degree of freedom system (exhibiting second order characteristics) with the response of a simple second order mass-spring system to obtain automatically the equivalent natural frequency and damping of the system. 2) Determining the best estimate of the present position and velocity of a ballistic missile by fitting the equations of motion to sampled positional radar data corrupted by gaussian noise. Here six parameters, position, and velocity, have to be continuously determined. Since it is an "on line" problem a high speed repetitive computer is necessary to allow the iteration process to converge before the arrival of the next data sample. 3) Boundary value problems in ordinary differential equations. Here the performance criteria are the errors in the satisfaction of the various boundary conditions applied. Since the number of boundary conditions is equal to the number of parameters (initial conditions) updated, the errors must all go to zero.

Other possible areas of application are: 1) Problems in the calculus of variations where to date only trial and error methods for the initial conditions of the Euler-Lagrange equations have been used to satisfy the boundary conditions. A typical such example is the mission profile problem[4] where it is desired to find the flight path which minimizes the amount of fuel or time it takes to attain a given point in space with a given velocity. 2) Solution of partial differential equations by serial techniques (i.e. where time is incremented and the space variable is taken to be the continuous variable) appears to be feasible since one of the main stumbling blocks in such an approach has also been the satisfaction of the boundary conditions.

Formulation of Problem

Let the physical system to be optimized on the computer be described by the following set of R first order differential equations

$$\dot{y}_r(t) = f_r(t, y_1(t), y_2(t), \ldots, y_R(t); \alpha_1, \alpha_2, \ldots, \alpha_N)$$

$$r = 1, \ldots, R \qquad (1)^*$$

where the α_n $n=1,\ldots,N$ are the parameters to be adjusted such that the system outputs, y_r $r=1,\ldots,R$, will have the desired performance characteristics. Let the performance criteria be described by error functions

$$\varepsilon_i(t) = \varepsilon_i\left[t, y_1(t), y_2(t), \ldots, y_R(t)\right] \quad i=1,\ldots,I \qquad (2)$$

(which vanish when the performance criteria are satisfied.)

Consider first the case where the system outputs are sampled at times t_j $j=0,1,\ldots,J$.** This represents J+1 points in time over which Eqs. (1) are fitted. Define

$$\varepsilon_i(t_j) \equiv \varepsilon_{ij} \qquad i=1,\ldots,I \qquad j=0,1,\ldots,J \qquad (3)$$

Let

$$S = \sum_{j=0}^{J} \sum_{i=1}^{I} G_{ij}\varepsilon_{ij}^2 \qquad G_{ij} = \text{constant} > 0 \qquad (4)$$

$$i=1,\ldots,I \qquad j=0,1,\ldots,J$$

denote a weighted sum of the errors squared which is to be minimized with respect to the N parameters α_n. To minimize S, Eq. (4) is differentiated with respect to α_n and the derivatives are set to zero.

$$0 = \frac{\partial S}{\partial \alpha_n} = 2 \sum_{j=0}^{J} \sum_{i=1}^{I} G_{ij}\varepsilon_{ij} \frac{\partial \varepsilon_{ij}}{\partial \alpha_n} \qquad n=1,\ldots,N \qquad (5)$$

Eq. (5) represents N functional equations for the N parameters, α_n, which are to be determined. To solve Eq. (5) on the analog computer the following iterative process is employed.

* $\dot{y}_r \equiv \frac{dy_r}{dt}$

** In "on line" problems this represents a comb of J+1 points. The comb moves up in time as new data samples arrive, i.e., the system is fitted over the latest J+1 sample points.

$$\alpha_n^{(k+1)} = \alpha_n^{(k)} - \frac{\sum_{j=0}^{J}\sum_{i=1}^{I} G_{ij}\varepsilon_{ij}^{(k)} \frac{\partial\varepsilon_{ij}^{(k)}}{\partial\alpha_n}}{\sum_{j=0}^{J}\sum_{i=1}^{I} G_{ij}\left(\frac{\partial\varepsilon_{ij}^{(k)}}{\partial\alpha_n}\right)^2} \qquad (6)^*$$

$n=1,\ldots,N$

where the superscript k denotes the iteration step. The iteration process described above is a "single step process," i.e., only one α_n is updated at a time. Thus, during the next cycle of computer operation the parameter α_{n+1} is updated, and so on. This has the advantage computer wise that Eq. (6) and the circuit for the partial derivatives

$$\frac{\partial\varepsilon_i}{\partial\alpha_n},$$

have to be instrumented once and not N times - with rotary stepping switches switching the circuit cyclically with respect to the index n during the reset period of each repetitive cycle, (see Fig. 2). Note that the iteration must stop, i.e. $\alpha_n^{(k+1)} = \alpha_n^{(k)}$, when Eq. (5) is satisfied. The denominator is a normalization factor and may be considered to be an "automatic gain control" to insure stability of the iteration process.

In the case of continuously sampled systems the iteration procedure becomes

$$\alpha_n^{(k+1)} = \alpha_n^{(k)} - \frac{\int_{t_o}^{T}\sum_{i=1}^{I} G_i(t)\varepsilon_i^{(k)}(t)\frac{\partial\varepsilon_i^{(k)}}{\partial\alpha_n}\,dt}{\int_{t_o}^{T}\sum_{i=1}^{I} G_i(t)\left[\frac{\partial\varepsilon_i^{(k)}}{\partial\alpha_n}\right]^2 dt} \qquad (7)$$

$n=1,\ldots,N$

and the same remarks apply.

Geometrical Motivation for the Iteration Process

For the purpose of this discussion it will be convenient to first simplify the notation by relabeling the (J+1)I errors, ε_{ij}, to ε_h $h=1,2,\ldots,H=(J+1)I$, and similarly the weighting factors. The double summation in Eqs. (4)-(6) then reduces to a single summation, and the error measure, S, becomes

$$S = \sum_{h=1}^{H} g_h\varepsilon_h^2 \qquad g_h > 0 \qquad h=1,2,\ldots,H \qquad (8)^*$$

The H errors $\varepsilon_1,\varepsilon_2,\ldots,\varepsilon_H$ depend on the N parameters, α_n. (For model fitting $N < H$, while for boundary value problems N=H and the errors should become zero.) Since it is desired to obtain a single step iteration procedure which will minimize Eq. (8), it is only necessary to consider one parameter, say α. Let

$$\vec{\varepsilon}(\alpha) = \left[\varepsilon_1(\alpha),\varepsilon_2(\alpha),\ldots,\varepsilon_H(\alpha)\right] \qquad (9)$$

denote the error vector before α is updated, and $\vec{\varepsilon}(\alpha+\delta\alpha)$ the error vector after α has been updated. To obtain a "steepest descent" path one tries to make the updated vector, $\vec{\varepsilon}(\alpha+\delta\alpha)$, orthogonal under the metric, g_h, to the derivative of the $\vec{\varepsilon}(\alpha)$ vector, with respect to the parameter α. This orthogonality condition can be expressed in equation form as the following scalar product:

$$\sum_{h=1}^{H} g_h\frac{\partial\varepsilon_h}{\partial\alpha}\varepsilon_h(\alpha+\delta\alpha) = 0 \qquad (10)$$

The object is now to solve Eq. (10) for $\delta\alpha$, which is the increment by which the parameter α is to be updated. Expanding $\varepsilon_h(\alpha+\delta\alpha)$ into a Taylor series including only first order terms, Eq. (10) becomes

*An investigation of the convergence of this iteration procedure is given in the Appendix.

*Shapiro[3] considers a more general formulation. The error measure, S, is defined as the positive definite quadratic

$$S = \sum_{\ell,h=1}^{H} g_{\ell h}\varepsilon_\ell\varepsilon_h$$

This formulation may be necessary in some problems involving noisy measurements. In the case of additive gaussian white noise this is the method of "Maximum Likelihood" if the inverse of the covariance matrix is chosen to be the matrix of the weighting factors, (g_{lh}). The corresponding iteration procedure is

$$\delta\alpha = -\frac{\sum_{\ell,h=1}^{H} g_{\ell h}\varepsilon_\ell\frac{\partial\varepsilon_h}{\partial\alpha}}{\sum_{\ell,h=1}^{H} g_{\ell h}\frac{\partial\varepsilon_\ell}{\partial\alpha}\frac{\partial\varepsilon_h}{\partial\alpha}}$$

By choosing $g_{\ell h} = \delta_{\ell h}g_h$, where $\delta_{\ell h}$ is the Kronecker delta function, this formulation reduces to the one presented in the text.

$$\sum_{h=1}^{H} g_h \frac{\partial \varepsilon_h}{\partial \alpha}\left[\varepsilon_h(\alpha) + \delta\alpha \frac{\partial \varepsilon_h}{\partial \alpha}\right] = 0 \tag{11}$$

Solving now Eq. (11) for $\delta\alpha$ yields the desired iteration formula

$$\delta\alpha = -\frac{\sum_{h=1}^{H} g_h \varepsilon_h \frac{\partial \varepsilon_h}{\partial \alpha}}{\sum_{h=1}^{H} g_h \left(\frac{\partial \varepsilon_h}{\partial \alpha}\right)^2} \tag{12}$$

In the case where the ε_h depend linearly on the parameters, Eqs. (10) and (11) are exactly equivalent. It follows that the iteration process must converge. Consider the vector diagram in Fig. (3). By Eq. (11) the vector

$$\vec{\varepsilon}(\alpha) + \delta\alpha \frac{\partial \vec{\varepsilon}}{\partial \alpha}$$

is orthogonal to the vector

$$\frac{\partial \vec{\varepsilon}}{\partial \alpha}.$$

Since the leg of a right triangle is smaller than the hypotenuse it follows immediately that

$$\vec{\varepsilon}(\alpha + \delta\alpha) \approx \vec{\varepsilon}(\alpha) + \delta\alpha \frac{\partial \vec{\varepsilon}}{\partial \alpha} \leq \vec{\varepsilon}(\alpha) \tag{13}$$

which was to be shown. The equality in Eq. (13) can only hold if $\delta\alpha = 0$ which means that the process has become stationary; if

$$\frac{\partial \vec{\varepsilon}}{\partial \alpha} \equiv 0$$

the system under consideration is independent of the parameter α, and hence α should be discarded. This is accomplished on the computer by implementing a zero detector for the quantity

$$\sum_{h=1}^{H} g_h \left(\frac{\partial \varepsilon_h}{\partial \alpha}\right)^2.$$

Similarly it is easy to show that S is reduced. Let $S(\alpha + \delta\alpha)$ denote the error measure after α has been updated, then

$$S(\alpha+\delta\alpha) = \sum_{h=1}^{H} g_h \left(\varepsilon_h + \delta\alpha \frac{\partial \varepsilon_h}{\partial \alpha}\right)^2$$

$$= \sum_{h=1}^{H} g_h \varepsilon_h^2 + 2\sum_{h=1}^{H} g_h \frac{\partial \varepsilon_h}{\partial \alpha}\left(\varepsilon_h + \delta\alpha \frac{\partial \varepsilon_h}{\partial \alpha}\right)\delta\alpha$$

$$- \sum_{h=1}^{H} g_h \left(\frac{\partial \varepsilon_h}{\partial \alpha}\right)^2 \delta\alpha^2$$

$$S(\alpha+\delta\alpha) = S(\alpha) + 0 - \sum_{h=1}^{H} g_h \left(\frac{\partial \varepsilon_h}{\partial \alpha}\right)^2 \delta\alpha^2$$

Since the g_h are all positive it follows immediately that

$S(\alpha + \delta\alpha) \leq S(\alpha)$.

If the ε_h are nonlinear this will still be true if $\delta\alpha$ is sufficiently small. In order to insure this, one can introduce an attenuation function as shown in Fig. (4). This may not be necessary if the ε_h are "well behaved."* Of course, caution must be exercised in nonlinear problems, for unless the ε_h are convex functions of the parameters, more than one minimum may exist and a good ball park guess for the initial setting of the parameters is necessary to converge to the absolute minimum.

The original motivation for the iteration procedure described by Eq. (12) was an attempt to simplify the Newton-Raphson which in this case is

$$\delta\alpha = -\frac{\sum_{h=1}^{H} g_h \varepsilon_h \frac{\partial \varepsilon_h}{\partial \alpha}}{\sum_{h=1}^{H} g_h \frac{\partial}{\partial \alpha}\left(\varepsilon_h \frac{\partial \varepsilon_h}{\partial \alpha}\right)} \tag{14}$$

The two iteration processes differ in that the denominator in Eq. (14) contains second derivative terms,

$$\frac{\partial^2 \varepsilon_h}{\partial \alpha^2},$$

which are ignored in Eq. (12). To obtain the second derivatives on the analog computer by the parameter influence coefficient technique requires roughly speaking, triple the equipment complement required by the original system (to obtain the first derivatives,

$$\frac{\partial \varepsilon_h}{\partial \alpha},$$

takes about double the complement - which is bad enough.) If the ε_h are linear functions of the parameters the two iteration processes are identical since then the second derivatives

*For the meaning of "well behaved" see the convergence proof in the Appendix.

are identically zero. In nonlinear problems it might happen that due to the second derivatives, the denominator in Eq. (14) can change sign in which case the process becomes unstable. This cannot happen with the proposed iteration procedure, Eq. (12), since there the denominator is always positive. Thus, both from a practical and theoretical point of view, the iteration procedure, Eq. (12), is superior to the Newton-Raphson scheme.

Illustrative Example

An interesting application to an actual problem* is the following three degree of freedom study, where it is desired to fit the response of an underwater body (in the pitch plane) with the response of a second order mass-spring system to determine the equivalent natural frequency and damping of the underwater body. The differential equations of motion of the underwater body simulated on the computer are (employing standard terminology)

$$\dot{w} = \frac{1}{m - \frac{\rho}{2}\ell^3 w}\left[muq + \frac{\rho}{2}\ell^4\left[Z'_{\dot{q}}\dot{q} + Z'_{qq}\, q|q|\right]\right. \tag{15}$$

$$+ \frac{\rho}{2}\ell^3\left[Z'_q uq + Z'_{wq}\, wq\right] + \frac{\rho}{2}\ell^2\left[Z'_* u^2 + Z'_w uw + Z'_{ww}\, w|w|\right.$$

$$\left.\left. + Z'_{\delta s}u^2(\delta_s + \alpha_s) + Z'_{\delta\delta s}u^2\ \delta_s|\delta_s|\right]\right]$$

$$\dot{q} = \frac{1}{I_y - \frac{\rho}{2}\ell^5 M'_{\dot{q}}}\left[\frac{\rho}{2}\ell^5 M'_{qq}\, q|q| + \frac{\rho}{2}\ell^4\left[M'_{\dot{w}}\dot{w} + M'_q uq\right.\right.$$

$$\left. + M'_{wq}wq\right] + \frac{\rho}{2}\ell^3\left[M'_* u^2 + M'_w uw + M'_{ww}\, w|w|\right. \tag{16}$$

$$\left.\left. + M'_{\delta s}u^2(\delta_s + \alpha_s) + M'_{\delta\delta s}u^2\ \delta_s|\delta_s|\right] + B_{zB}\sin\theta\right]$$

$$\dot{\theta} = q \tag{17}$$

$$\dot{\alpha}_s = k_\alpha\left(\frac{w - x_s q}{u}\right) \tag{18}$$

*This problem has been conducted for the David Taylor Model Basin at the Princeton Computation Center, Electronic Associates, Inc., by Mr. Paul Landauer; the author wishes to thank the David Taylor Model Basin for permission to use this example.

where

u,v,w	are the velocities along the body axes x,y,z
q	is the pitch rate
α	is the angle of attack
θ	is the angle of elevation

The equation of the model approximating the underwater body is

$$\ddot{\theta}' + 2\zeta\omega\dot{\theta}' + \omega^2\theta' = 0 \tag{19}$$

Eqs. (15) to (19) together with the following are instrumented on the computer.

$$\alpha^{(k+1)} = \alpha^{(k)} - \frac{\int_0^T \varepsilon^{(k)}\lambda^{(k)}dt}{\int_0^T \left(\lambda^{(k)}\right)^2 dt} \quad \text{where } \alpha = \begin{bmatrix}\omega \\ \zeta\end{bmatrix} \tag{20}$$

$$\ddot{\lambda} + 2\zeta\omega\dot{\lambda} + \omega^2\lambda = \begin{bmatrix}-2(\zeta\dot{\theta}' + \omega\theta') \\ -2\omega\dot{\theta}'\end{bmatrix} \tag{21}$$

$$\text{where } \lambda = \begin{bmatrix}\frac{\partial\theta'}{\partial\omega} \\ \frac{\partial\theta'}{\partial\zeta}\end{bmatrix}$$

$$\varepsilon(t) = \theta'(t) - \theta(t) \tag{22}$$

The terms in the upper part of the brackets are used when updating ω, and in the lower part when updating ζ. Thus the same circuitry with minor switching is employed to update alternately ω and ζ. See Fig. (5). It was found that convergence is more rapid in the underdamped than in the overdamped use. This is so because

$$\int_0^T \varepsilon^2 dt$$

as a function of the parameters ω and ζ has a much shallower bottom in the overdamped case, and therefore is less sensitive. The problem was simulated on a real time computer where the hold and reset relays were externally controlled for automatic cycling and sampling purposes. An average of eight runs (or four complete iteration cycles) were needed for convergence, and each run was only of the order of one second.

Boundary Value Problems

The analog computer has little difficulty in solving most initial value problems. Boundary value problems, however, present a different story. The case where both the differential equations and the boundary conditions are linear has been treated in detail by Yanowitch[5]. In that event the principle of superposition holds and a linear combination of independent solutions (each of the solutions satisfying the boundary conditions at the initial point) can be found such that it satisfies the boundary conditions at all the points where they are prescribed.

This, however, involves a matrix inversion, which, if there are more than two boundary conditions to be satisfied in addition to the conditions at the initial point, implies an inordinate number of multipliers (and point storage devices for boundary value problems with conditions at more than two points.)

The other problem is when either the differential equations or the boundary conditions or both are nonlinear. Superposition no longer holds and hence, different techniques must be applied. The method proposed again employs the iteration scheme given by Eq. (6).

$$\alpha_n^{(k+1)} = \alpha_n^{(k)} - \frac{\sum_{j=0}^{J}\sum_{i=1}^{I} \varepsilon_{ij}^{(k)} \frac{\partial \varepsilon_{ij}^{(k)}}{\partial \alpha_n}}{\sum_{j=0}^{J}\sum_{i=1}^{I}\left(\frac{\partial \varepsilon^{(k)}}{\partial \alpha_n}\right)^2} \tag{6a}$$

$$n=1,2,\ldots,(J+1)I$$

$$G_{ij} \equiv 1 \quad \text{all } i,j$$

with the interpretation that the (J+1)I boundary conditions are prescribed at the points t_j $j=0,1,\ldots,J$; and the (J+1)I parameters, α_n, represent the initial conditions which must be determined at the initial end, t_o.

As an illustrative example, consider the very simple linear two point boundary value problem

$$\ddot{x} + a\dot{x} + bx = 0 \tag{23a}$$

with boundary conditions prescribed at t=0 and t=L

$$x(0) = x_o \tag{23b}$$

$$x(L) + c\dot{x}(L) = d \tag{23c}$$

In this particular case $\dot{x}_o$ is the parameter to be determined, such that Eq. (23c) is satisfied; the iteration formula Eq. (6a) reduces to

$$\dot{x}_o^{(k+1)} = \dot{x}_o^{(k)} - \frac{\varepsilon^{(k)}(L)}{\left.\frac{\partial \varepsilon^{(k)}}{\partial \dot{x}_o}\right|_{t=L}} \tag{24}$$

which in this particular case reduces to the Newton-Raphson scheme. Here

$$\varepsilon^{(k)}(L)$$

is defined by

$$\varepsilon^{(k)}(L) \equiv x^{(k)}(L) + c\dot{x}^{(k)}(L) - d \tag{25}$$

The partial derivative,

$$\left.\frac{\partial \varepsilon^{(k)}}{\partial \dot{x}_o}\right|_{t=L}$$

is obtained by the parameter influence coefficient technique. Define

$$\frac{\partial x}{\partial \dot{x}_o} \equiv u(t) \tag{26}$$

Then u(t) is determined by differentiating Eq. (23a) with respect to $\dot{x}_o$, yielding the differential equation

$$\ddot{u} + a\dot{u} + bu = 0 \tag{27}$$

with initial conditions

$$u(0) = 0 \qquad \dot{u}(0) = 1 \tag{28}$$

Eqs. (23a), (23b), (24)-(28) are instrumented on the analog computer. See Fig. (6). Since the example is linear with only one parameter to be updated, one might suspect that the procedure will converge in one iteration step. This is indeed true as can be easily verified analytically. (Note that Eq. (27) is independent of the iteration procedure, which would not be true in a nonlinear problem.) Formula (24) may also be obtained from the following consideration which gives more insight into the process. The boundary condition (23c) can be considered to be also a function of $\dot{x}_o$ and may therefore be written

$$x(\dot{x}_o,L) + c\dot{x}(\dot{x}_o,L) = d \tag{29}$$

Hence a variation of the solution at the boundary must satisfy

$$\left.\frac{\partial x}{\partial \dot{x}_o}\right|_{t=L} \delta\dot{x}_o + \delta x(L) \tag{30}$$

$$+ c\left(\left.\frac{\partial \dot{x}}{\partial \dot{x}_o}\right|_{t=\ell} \delta\dot{x}_o + \delta\dot{x}(L)\right) = 0$$

solving for $\delta\dot{x}_o$ in Eq. (30) yields

$$\delta \dot{x}_o = - \frac{\delta x(L) + c\delta \dot{x}(L)}{\left.\frac{\partial x}{\partial \dot{x}_o}\right|_{t=L} + c \left.\frac{\partial \dot{x}}{\partial \dot{x}_o}\right|_{t=L}} \tag{31}$$

The denominator of Eq. (31) is seen to be equal to

$$\left.\frac{\partial \mathcal{E}}{\partial \dot{x}_o}\right|_{t=L} ,$$

and replacing the numerator with $\mathcal{E}(L)$, the iteration process given by Eq. (24) is attained.

Conclusions

The technique presented in this paper for optimization and solution of boundary value problemt is felt to be general and applicable even to problems where a large number of parameters have to be determined. Although in smaller problems simpler schemes may be used, the chief advantage of the technique described lies in the convergence of the process under fairly general conditions. Some of the chief drawbacks and difficulties encountered are:

1. Equipment wise the scheme is expensive for anything but trivial problems since a "duplication" of the model equations is required for the parameter influence coefficients; the logic and sampling circuitry can be extensive if the number of conditions imposed and parameters to be determined is substantial. It is visualized that the logical functions in large problems be perhaps digitally instrumented. The fact that the equipment requirements are large should not be a dissuading factor since the problems considered are formidable.

2. The amplitude scaling of the parameter influence equations may require some trial and error adjustment in complex problems where a priori analysis is difficult.

3. A good "ball park" initial guess is important for convergence in non-linear problems, where several solutions may be possible.

Acknowledgement

For their advice, technical assistance, and encouragement I am thankful to my associates; I am particularly indebted to Romeo R. Favreau for his discriminating criticism, to Hans Witsenhausen, also of Electronic Associates, Inc., for pointing out the geometrical significance of the iteration procedure, and to Dr. Jack Walsh of Convair, San Diego for his help in investigating the convergence of the iteration procedure.

References

(1) Meissinger, H.F., "The Use of Parameter Influence Coefficients in Computer Analysis of Dynamic Systems." Proceedings Western Joint Computer Conference, San Francisco, May 1960.

(2) Margolis, M. and Leondes, C.T., "A Parameter Tracking Servo for Adaptive Control Systems," IRE Transactions on Automatic Control, Vol. AC-4, Number 2, November 1959, p. 100-111.

(3) Shapiro, I.I., "The Prediction of Ballistic Missile Trajectories from Radar Observations," McGraw Hill, 1957.

(4) Mengel, A.S., "Optimum Trajectories," Project Cyclone Symposium I on REAC Techniques," March 1951.

(5) Yanowitch, M., "The Solution of Boundary Value Problems on a REAC Analog Computer," Presented at the Association for Computing Machinery in Philadelphia on September 16, 1955.

Appendix

Investigation of the Convergence of the Iteration Procedure

The function, S, given by Eq. (4) and the iteration process, Eq. (6), which is to minimize S can be brought into the normalized form

$$S = \sum_{h=1}^{H} \left[\varepsilon_h(\alpha_1,\dots,\alpha_N)\right]^2$$

$$\alpha_n^{(k+1)} = \alpha_n^{(k)} - \frac{\sum_{h=1}^{H} \varepsilon_h(\alpha_1^{(k+1)},\dots,\alpha_{(n-1)}^{(k+1)},\alpha_n^{(k)},\dots,\alpha_N^{(k)}) \frac{\partial \varepsilon_h}{\partial \alpha_n}\Big|\left(\alpha_1^{(k+1)},\dots,\alpha_n^{(k)}\right)}{\sum_{h=1}^{I} \left[\frac{\partial \varepsilon_h}{\partial \alpha_n}\Big| \left(\alpha_1^{(k+1)},\dots,\alpha_N^{(k)}\right)\right]^2}$$

by defining

$$\varepsilon_h \equiv \sqrt{G_{ij}}\,\varepsilon_{ij}$$

where $h=1,2,\dots,H=(J+1)I$ as $i=1,2,\dots,I$ and $j=0,1,\dots,J$. Thus the double sum is reduced to a single summation.

The cases of one function of one variable, and two functions of two variables are considered. The arguments in the latter case with slight modifications hold for the general case.

Case I

$$S = \left[\varepsilon(\alpha)\right]^2$$

$$\alpha^{(k+1)} = \alpha^{(k)} - \frac{\varepsilon(\alpha^{(k)})}{\varepsilon'(\alpha^{(k)})} \qquad \text{where } \varepsilon'(\alpha^{(k)}) \equiv \frac{\partial \varepsilon}{\partial \alpha}\Big|_{\alpha=\alpha^{(k)}}$$

Define

$$\lambda \equiv \frac{\varepsilon(\alpha^{(k)})}{\varepsilon'(\alpha^{(k)})}$$

then

$$S' = 2\left[\varepsilon(\alpha)\right]\varepsilon'(\alpha) = 2\lambda\varepsilon'(\alpha)^2$$

Let

$$S^{(k)} \equiv \left[\varepsilon(\alpha^{(k)})\right]^2$$

and

$$S^{(k+1)} \equiv \left[\varepsilon(\alpha^{(k+1)})\right]^2$$

So

$$S^{(k+1)} = \left[\varepsilon(\alpha^{(k)}-\lambda)\right]^2$$

By the mean value theorem

$$\varepsilon(\alpha^{(k)}-\lambda) = \varepsilon(\alpha^{(k)}) - \varepsilon'(\bar{\alpha}^{(k)})\lambda \qquad \text{where } \alpha^{(k)}-\lambda \leq \bar{\alpha}^{(k)} \leq \alpha^{(k)}$$

and

$$S^{(k+1)} = \left[\varepsilon(\alpha^{(k)}) - \varepsilon'(\bar{\alpha}^{(k)})\lambda\right]^2$$

$$= \left[\varepsilon(\alpha^{(k)}) - \varepsilon'(\bar{\alpha}^{(k)}) \frac{\varepsilon(\alpha^{(k)})}{\varepsilon'(\alpha^{(k)})}\right]^2$$

$$= \left[\varepsilon(\alpha^{(k)})\right]^2 \left[1 - \frac{\varepsilon'(\bar{\alpha}^{(k)})}{\varepsilon'(\alpha^{(k)})}\right]^2$$

$$S^{(k+1)} = S^{(k)} \left[1 - \frac{\varepsilon'(\bar{\alpha}^{(k)})}{\varepsilon'(\alpha^{(k)})}\right]^2$$

If $\varepsilon(\alpha)$ is "well behaved" so that $\varepsilon'(\alpha)$ is continuous and the original estimate is "close enough," the requirement

$$0 < \frac{\varepsilon'(\bar{\alpha}^{(k)})}{\varepsilon'(\alpha^{(k)})} < 2$$

can be expected to hold. (The additional requirement that $\varepsilon'(\alpha) \neq 0$ where α is the value at which S has a minimum seems necessary, and is sufficient so that $\varepsilon'(\alpha^{(k)})$ and $\varepsilon'(\bar{\alpha}^{(k)})$ have the same sign, and the quotient,

$$\frac{\varepsilon'(\bar{\alpha}^{(k)})}{\varepsilon'(\alpha^{(k)})},$$

is meaningful. Otherwise $\varepsilon'(\alpha^{(k)})$ might be zero.) In such a case one has

$$S^{(k+1)} = S^{(k)}\delta \qquad \text{where} \quad 0 < \delta < 1$$

and so

$$S^{(k+1)} < S^{(k)}$$

Here it was tacitly assumed that $\lambda \neq 0$. If $\lambda = 0$, then $S' = 0$, and so S has a minimum at $\alpha^{(k)}$. Then $\alpha^{(k+1)} = \alpha^{(k)}$ and the process is stationary.

Case II

$$S = G(\alpha,\beta)^2 + F(\alpha,\beta)^2$$

Single step iteration described is considered. According to the scheme (subscript α denotes partial derivative by α, etc.),

$$\alpha^{(k+1)} = \alpha^{(k)} - \frac{G(\alpha^{(k)},\beta^{(k)})\, G_\alpha(\alpha^{(k)},\beta^{(k)}) + F(\alpha^{(k)},\beta^{(k)})\, F_\alpha(\alpha^{(k)},\beta^{(k)})}{G_\alpha(\alpha^{(k)},\beta^{(k)})^2 + F_\alpha(\alpha^{(k)},\beta^{(k)})^2}$$

$$= \alpha^{(k)} - \lambda$$

where

$$\lambda \equiv \frac{1}{2}\,\frac{S_\alpha}{G_\alpha^2 + F_\alpha^2}$$

Let

$$S^{(k)} \equiv G(\alpha^{(k)},\beta^{(k)})^2 + F(\alpha^{(k)},\beta^{(k)})^2$$

and

$$S^{(k+1)} \equiv G(\alpha^{(k+1)},\beta^{(k)})^2 + F(\alpha^{(k+1)},\beta^{(k)})^2$$

Then

$$S^{(k+1)} = G(\alpha^{(k)} - \lambda,\beta^{(k)})^2 + F(\alpha^{(k)} - \lambda,\beta^{(k)})^2$$

From the mean value theorem it follows

$$S^{(k+1)} = \left[G(\alpha^{(k)},\beta^{(k)}) - G_\alpha(\bar{\alpha}^{(k)},\beta^{(k)})\lambda\right]^2 + \left[F(\alpha^{(k)},\beta^{(k)}) - F_\alpha(\bar{\alpha}^{(k)},\beta^{(k)})\lambda\right]^2$$

where

$$\alpha^{(k)} - \lambda \leq \bar{\alpha}^{(k)} \leq \alpha^{(k)}$$

$$S^{(k+1)} = \left[G(\alpha^{(k)},\beta^{(k)}) - G_\alpha(\bar{\alpha}^{(k)},\beta^{(k)})\,\frac{G(\alpha^{(k)},\beta^{(k)})G_\alpha(\alpha^{(k)},\beta^{(k)}) + F(\alpha^{(k)},\beta^{(k)})F_\alpha(\alpha^{(k)},\beta^{(k)})}{G_\alpha(\alpha^{(k)},\beta^{(k)})^2 + F_\alpha(\alpha^{(k)},\beta^{(k)})^2}\right]^2$$

$$+\left[F(\alpha^{(k)},\beta^{(k)}) - F_\alpha(\bar{\alpha}^{(k)},\beta^{(k)})\,\frac{G(\alpha^{(k)},\beta^{(k)})G_\alpha(\alpha^{(k)},\beta^{(k)}) + F(\alpha^{(k)},\beta^{(k)})F_\alpha(\alpha^{(k)},\beta^{(k)})}{G_\alpha(\alpha^{(k)},\beta^{(k)})^2 + F_\alpha(\alpha^{(k)},\beta^{(k)})^2}\right]^2$$

$$= G(\alpha^{(k)},\beta^{(k)})^2\left[1 - \frac{G_\alpha(\bar{\alpha}^{(k)},\beta^{(k)})G_\alpha(\alpha^{(k)},\beta^{(k)}) + \frac{F(\alpha^{(k)},\beta^{(k)})}{G(\alpha^{(k)},\beta^{(k)})}\,F_\alpha(\alpha^{(k)},\beta^{(k)})G(\bar{\alpha}^{(k)},\beta^{(k)})}{G_\alpha(\bar{\alpha}^{(k)},\beta^{(k)})^2 + F_\alpha(\alpha^{(k)},\beta^{(k)})^2}\right]^2$$

$$+ F(\alpha^{(k)},\beta^{(k)})^2\left[1 - \frac{F_\alpha(\bar{\alpha}^{(k)},\beta^{(k)})F_\alpha(\alpha^{(k)},\beta^{(k)}) + \frac{G(\alpha^{(k)},\beta^{(k)})}{F(\alpha^{(k)},\beta^{(k)})}\,F_\alpha(\bar{\alpha}^{(k)},\beta^{(k)})G_\alpha(\alpha^{(k)},\beta^{(k)})}{G_\alpha(\alpha^{(k)},\beta^{(k)})^2 + F_\alpha(\alpha^{(k)},\beta^{(k)})^2}\right]^2$$

By the generalized mean value theorem

$$\frac{G(\alpha^{(k)},\beta^{(k)})}{F(\alpha^{(k)},\beta^{(k)})} = \frac{G_\alpha(\bar{\bar{\alpha}}^{(k)},\beta^{(k)})}{F_\alpha(\bar{\bar{\alpha}}^{(k)},\beta^{(k)})}$$

and so

$$S^{(k+1)} = G(\alpha^{(k)},\beta^{(k)})^2 \left[1 - \frac{G_\alpha(\bar{\alpha}^{(k)},\beta^{(k)})G_\alpha(\alpha^{(k)},\beta^{(k)}) + \frac{G_\alpha(\bar{\alpha}^{(k)},\beta^{(k)})}{G_\alpha(\bar{\bar{\alpha}}^{(k)},\beta^{(k)})} F_\alpha(\alpha^{(k)},\beta^{(k)})F_\alpha(\bar{\bar{\alpha}}^{(k)},\beta^{(k)})}{G_\alpha(\alpha^{(k)},\beta^{(k)})^2 + F_\alpha(\alpha^{(k)},\beta^{(k)})^2} \right]^2$$

$$+ F(\alpha^{(k)},\beta^{(k)})^2 \left[1 - \frac{F_\alpha(\bar{\alpha}^{(k)},\beta^{(k)})F_\alpha(\alpha^{(k)},\beta^{(k)}) + \frac{F_\alpha(\bar{\alpha}^{(k)},\beta^{(k)})}{F_\alpha(\bar{\bar{\alpha}}^{(k)},\beta^{(k)})} G_\alpha(\alpha^{(k)},\beta^{(k)})G_\alpha(\bar{\bar{\alpha}}^{(k)},\beta^{(k)})}{G_\alpha(\alpha^{(k)},\beta^{(k)})^2 + F_\alpha(\alpha^{(k)},\beta^{(k)})^2} \right]^2$$

Again, if F and G are "well behaved" and the first estimate is "close enough," then

$$\frac{G_\alpha(\bar{\alpha}^{(k)},\beta^{(k)})}{G_\alpha(\bar{\bar{\alpha}}^{(k)},\beta^{(k)})}, \quad \frac{F_\alpha(\bar{\alpha}^{(k)},\beta^{(k)})}{F_\alpha(\bar{\bar{\alpha}}^{(k)},\beta^{(k)})}, \quad \frac{G_\alpha(\bar{\alpha}^{(k)},\beta^{(k)})G_\alpha(\alpha^{(k)},\beta^{(k)}) + F_\alpha(\alpha^{(k)},\beta^{(k)})F_\alpha(\bar{\bar{\alpha}}^{(k)},\beta^{(k)})}{G_\alpha(\alpha^{(k)},\beta^{(k)})^2 + F_\alpha(\alpha^{(k)},\beta^{(k)})^2},$$

and

$$\frac{F_\alpha(\bar{\alpha}^{(k)},\beta^{(k)})F_\alpha(\alpha^{(k)},\beta^{(k)}) + G_\alpha(\alpha^{(k)},\beta^{(k)})G_\alpha(\bar{\bar{\alpha}}^{(k)},\beta^{(k)})}{G_\alpha(\alpha^{(k)},\beta^{(k)})^2 + F_\alpha(\alpha^{(k)},\beta^{(k)})^2}$$

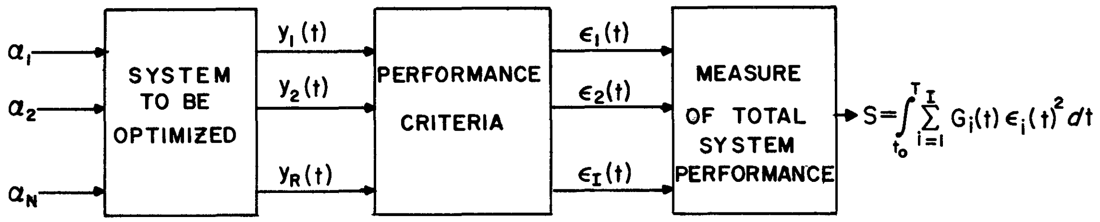
are close to unity, so that

$$S^{(k+1)} = G(\alpha^{(k)},\beta^{(k)})^2 \delta_1 + F(\alpha^{(k)},\beta^{(k)})^2 \delta_2 \quad \text{where} \quad 0 < \delta_1, \delta_2 < 1$$

and hence,

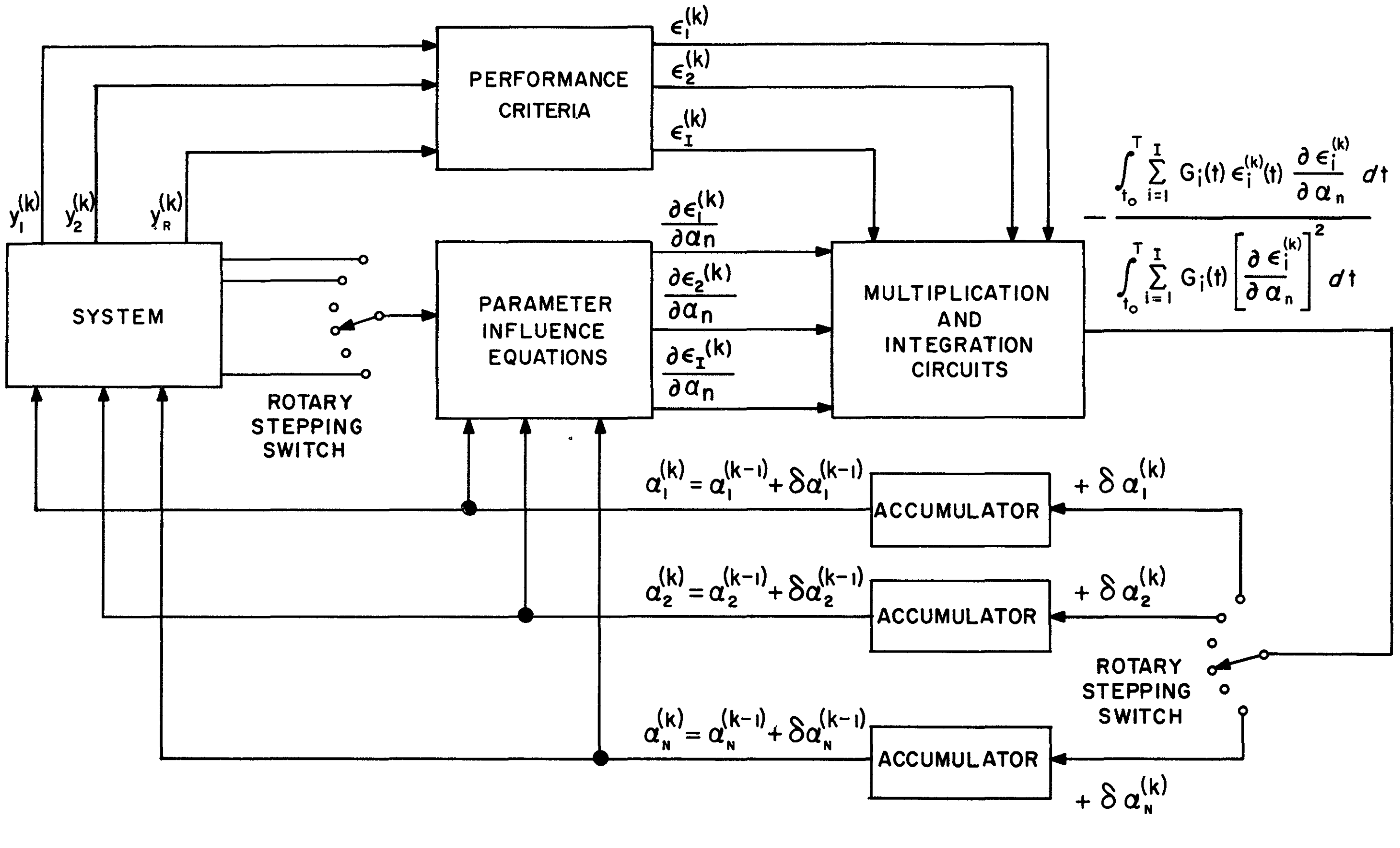
$$S^{(k+1)} < S^{(k)} .$$

In order that the quotients of partials be meaningful it is sufficient to assume that G_α and F_α are not zero at the minimum value of S. (Note, if they both did vanish, it would imply that the system is independent of α.) Again, one proceeded under the tacit assumption that $\lambda \neq 0$. If $\lambda = 0$, then $\alpha^{(k+1)} = \alpha^{(k)}$ and so the process is stationary.

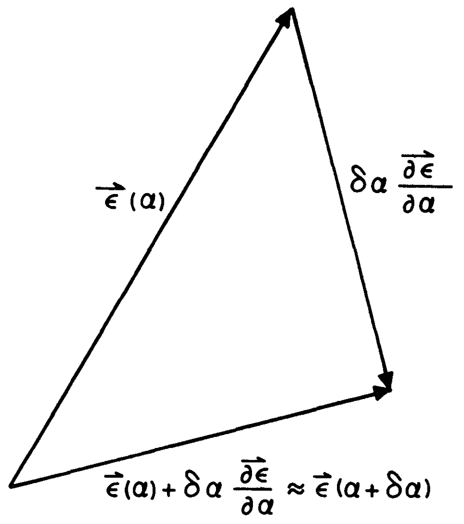


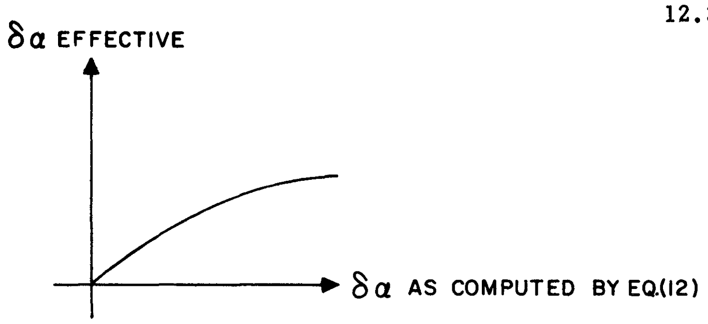
WHERE α_n	$n = 1,2,----,N$	ARE THE ADJUSTABLE SYSTEM PARAMETERS
$y_r(t)$	$r = 1,2,----,R$	ARE THE SYSTEM OUTPUTS
$\epsilon_i(t)$	$i = 1,2,----,I$	ARE ERRORS IN THE SATISFACTION OF THE PERFORMANCE CRITERIA
$G_i(t)$	$i = 1,2,-----,I$	ARE POSITIVE WEIGHTING FUNCTIONS

OPTIMIZATION PROBLEM
FIGURE (1)


PERFORMANCE CRITERIA
$\epsilon_1^{(k)}$
$\epsilon_2^{(k)}$
$\epsilon_I^{(k)}$
$y_1^{(k)}$
$y_2^{(k)}$
$y_R^{(k)}$
SYSTEM
ROTARY STEPPING SWITCH
PARAMETER INFLUENCE EQUATIONS
$\frac{\partial\epsilon_1^{(k)}}{\partial\alpha_n}$
$\frac{\partial\epsilon_2^{(k)}}{\partial\alpha_n}$
$\frac{\partial\epsilon_I^{(k)}}{\partial\alpha_n}$
MULTIPLICATION AND INTEGRATION CIRCUITS
$-\frac{\int_{t_0}^{T}\sum_{i=1}^{I} G_i(t)\,\epsilon_i^{(k)}(t)\,\frac{\partial\epsilon_i^{(k)}}{\partial\alpha_n}\,dt}{\int_{t_0}^{T}\sum_{i=1}^{I} G_i(t)\left[\frac{\partial\epsilon_i^{(k)}}{\partial\alpha_n}\right]^2 dt}$
$\alpha_1^{(k)}=\alpha_1^{(k-1)}+\delta\alpha_1^{(k-1)}$
ACCUMULATOR
$+\,\delta\,\alpha_1^{(k)}$
$\alpha_2^{(k)}=\alpha_2^{(k-1)}+\delta\alpha_2^{(k-1)}$
ACCUMULATOR
$+\,\delta\,\alpha_2^{(k)}$
ROTARY STEPPING SWITCH
$\alpha_N^{(k)}=\alpha_N^{(k-1)}+\delta\alpha_N^{(k-1)}$
ACCUMULATOR
$+\,\delta\,\alpha_N^{(k)}$

AUTOMATIC LEAST SQUARES OPTIMIZATION SCHEME
FIGURE (2)


$\vec{\epsilon}(\alpha)$
$\delta\alpha \frac{\partial\vec{\epsilon}}{\partial\alpha}$
$\vec{\epsilon}(\alpha)+\delta\alpha\frac{\partial\vec{\epsilon}}{\partial\alpha} \approx \vec{\epsilon}(\alpha+\delta\alpha)$



$\delta\alpha$ ATTENUATION FOR NONLINEAR PROBLEMS

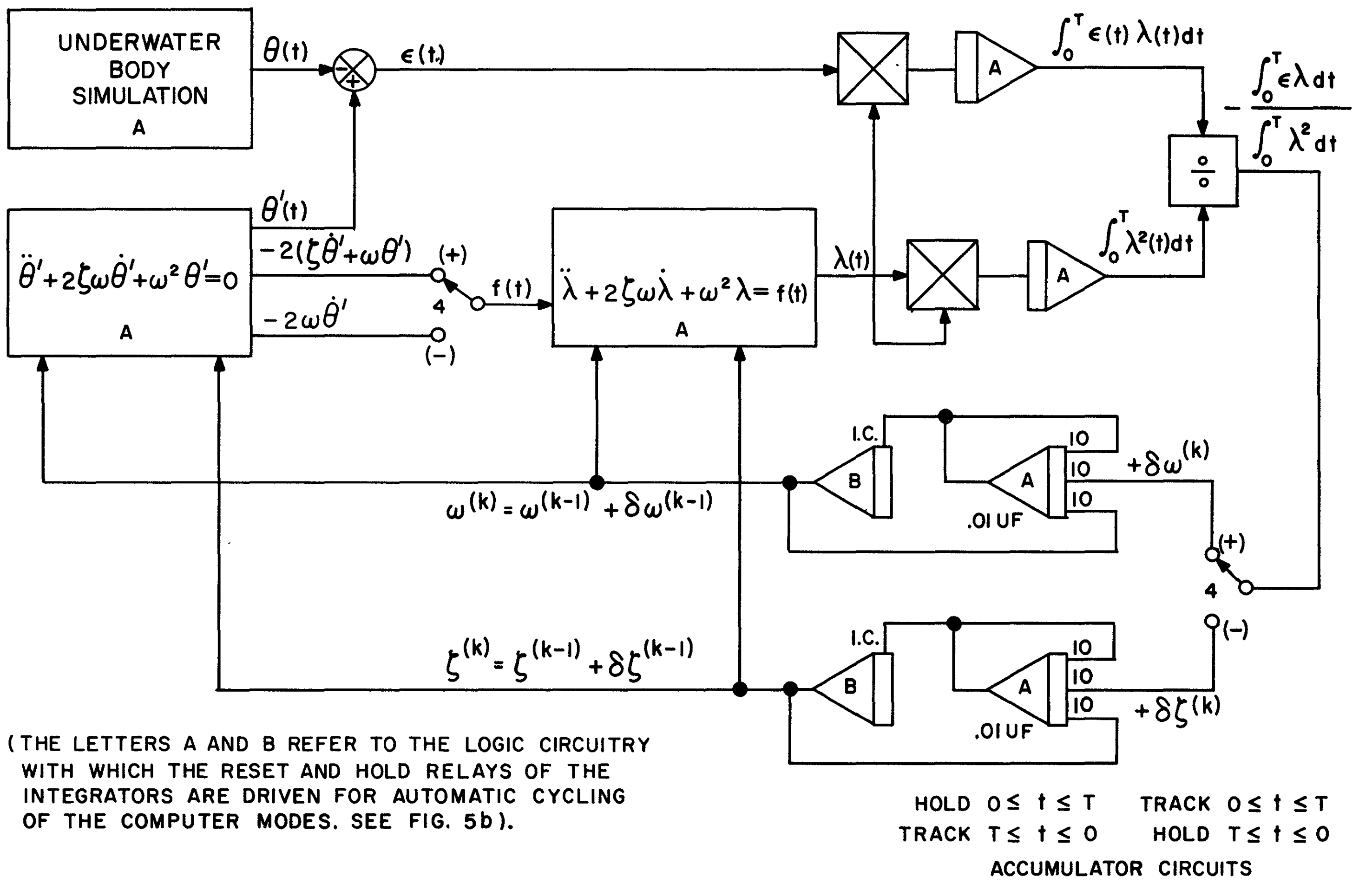
FIGURE (4)

FROM THE ORTHOGONALITY OF THE VECTORS $\frac{\partial\vec{\epsilon}}{\partial\alpha}$ AND $\vec{\epsilon}(\alpha)+\delta\alpha\frac{\partial\vec{\epsilon}}{\partial\alpha}$, IT

FOLLOWS THAT $\vec{\epsilon}(\alpha+\delta\alpha) \leq \vec{\epsilon}(\alpha)$.

VECTOR DIAGRAM OF ITERATION PROCESS

FIGURE (3)

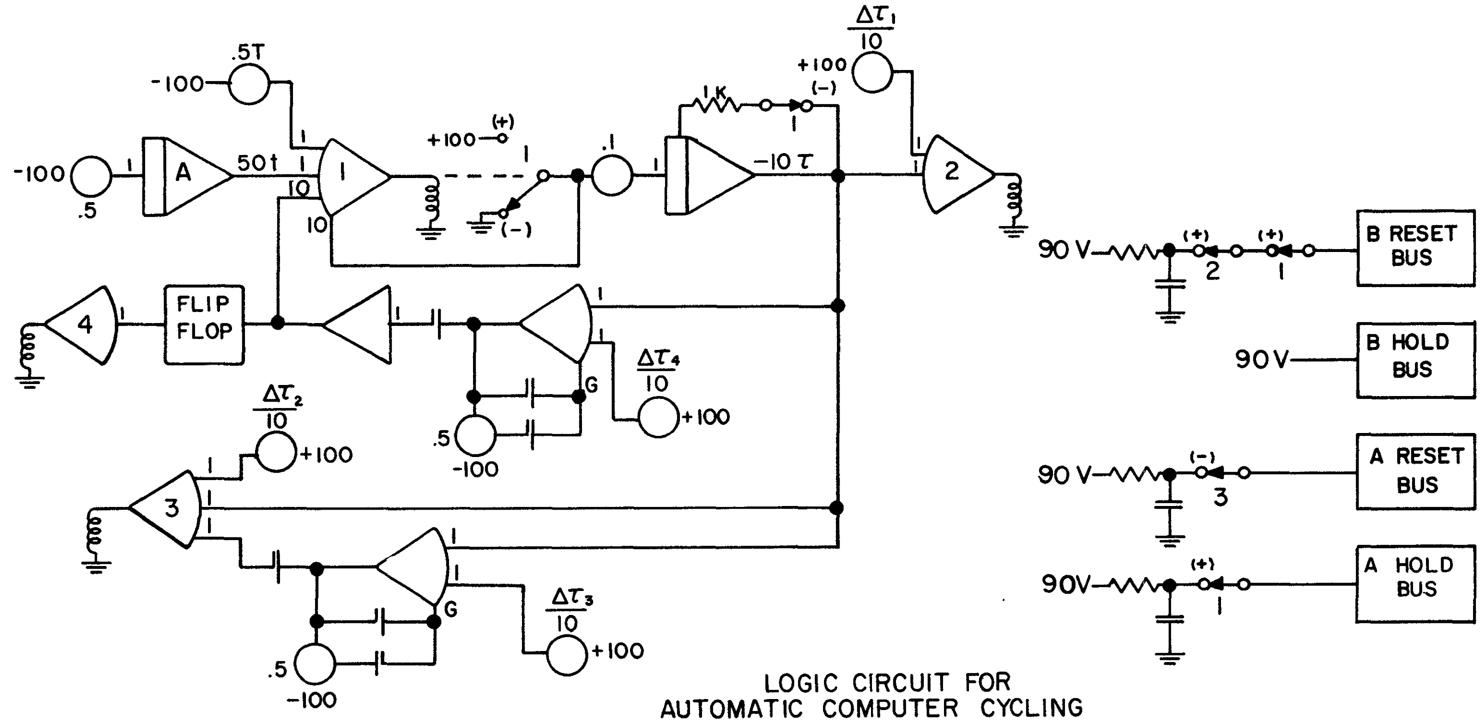

UNDERWATER BODY SIMULATION A
$\theta(t)$
$\epsilon(t)$
$\int_0^T \epsilon(t)\lambda(t)dt$
$-\frac{\int_0^T \epsilon\lambda dt}{\int_0^T \lambda^2 dt}$
$\theta'(t)$
$\ddot{\theta}'+2\zeta\omega\dot{\theta}'+\omega^2\theta'=0$ A
$-2(\zeta\dot{\theta}'+\omega\theta')$
$-2\omega\dot{\theta}'$
(+) 4 (−)
f(t)
$\ddot{\lambda}+2\zeta\omega\dot{\lambda}+\omega^2\lambda=f(t)$ A
$\lambda(t)$
$\int_0^T \lambda^2(t)dt$
A
I.C.
B
10 10 10
.01 UF
$+\delta\omega^{(k)}$
$\omega^{(k)}=\omega^{(k-1)}+\delta\omega^{(k-1)}$
$\zeta^{(k)}=\zeta^{(k-1)}+\delta\zeta^{(k-1)}$
$+\delta\zeta^{(k)}$
HOLD $0\leq t\leq T$ TRACK $0\leq t\leq T$
TRACK $T\leq t\leq 0$ HOLD $T\leq t\leq 0$
ACCUMULATOR CIRCUITS

(THE LETTERS A AND B REFER TO THE LOGIC CIRCUITRY WITH WHICH THE RESET AND HOLD RELAYS OF THE INTEGRATORS ARE DRIVEN FOR AUTOMATIC CYCLING OF THE COMPUTER MODES. SEE FIG. 5b).

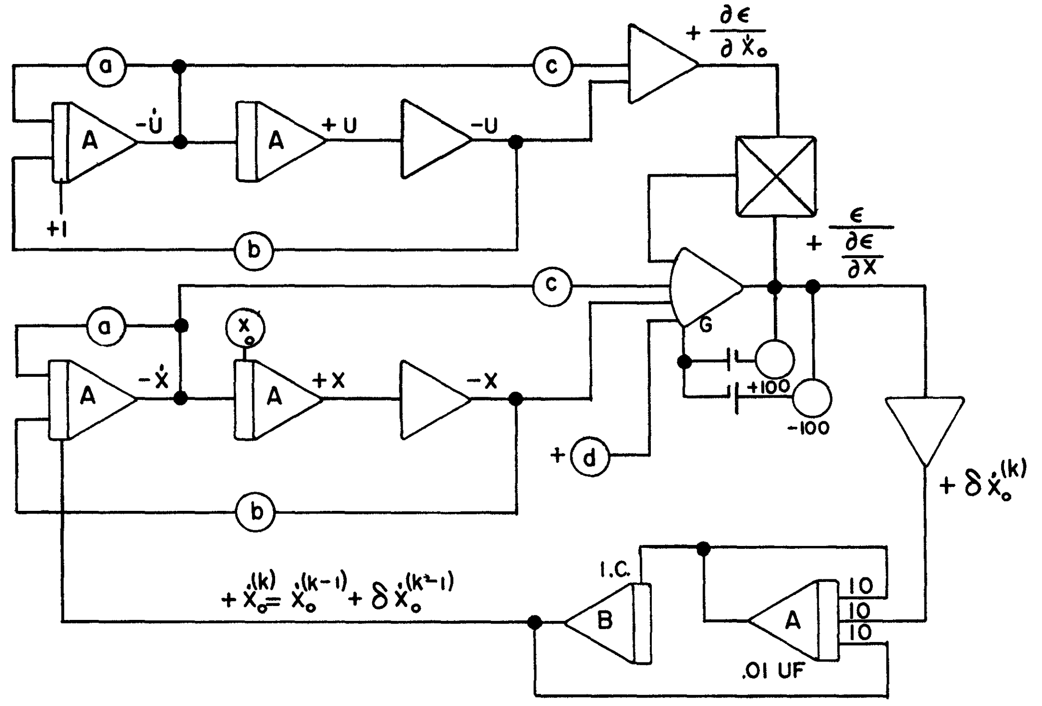
DYNAMIC CURVE FITTING OF UNDERWATER BODY RESPONSE

FIGURE (5a)

CYCLE	RELAY A	RESET	HOLD	RELAY B	RESET	HOLD	TIME
1	OPERATE	0	0	HOLD	0	1	T
2	HOLD	0	1	RESET	1	1	$\Delta\tau_1$ $\Delta\tau_2$ $\Delta\tau_3$ $\Delta\tau_4$
3	HOLD	0	1	HOLD	0	1	
4	RESET	1	1	HOLD	0	1	
5	HOLD	0	1	HOLD	0	1	
1	OPERATE	0	0	HOLD	0	1	T



LOGIC CIRCUIT FOR
AUTOMATIC COMPUTER CYCLING
FIGURE (5b)


a
c
$+\frac{\partial \epsilon}{\partial \dot{X}_o}$
A
$-\dot{U}$
A
+U
-U
+1
b
$+\frac{\epsilon}{\frac{\partial \epsilon}{\partial X}}$
c
a
$\dot{X}_o$
G
A
$-\dot{X}$
A
+X
-X
+100
-100
+d
b
$+\delta \dot{X}_o^{(k)}$
I.C.
B
A
10
10
10
.01 UF
$+\dot{X}_o^{(k)} = \dot{X}_o^{(k-1)} + \delta \dot{X}_o^{(k-1)}$

TWO POINT BOUNDARY VALUE PROBLEM $\ddot{X} + a\dot{X} + bX = 0$ $X(0) = X_o$
$X(L) + C\dot{X}(L) = d$

FIGURE (6)

ANALOG SIMULATION OF UNDERGROUND WATER FLOW IN THE LOS ANGELES COASTAL PLAIN

by

Donald A. Darms
EAI Computation Center

and

Howell N. Tyson*
IBM Corporation

Summary

A general purpose electronic differential analyzer approach to the study of the underground water flow in the Los Angeles Coastal Plain is described. The structure of the Los Angeles Coastal Plain aquifer system is discussed. A suitably simplified model of the aquifer system is proposed leading to the diffusion equation in two dimensions. The diffusion equation model is approximated by a system of difference-differential equations defined at a set of asymmetrically spaced node points. An electronic differential analyzer circuit is chosen which retains a one-to-one correspondence, in certain of its features, to the asymmetric network approximation. Some special checking techniques are discussed.

Introduction

The decline of California's ground-water tables in recent years represents a serious threat to the continued growth of the state. Of particular concern is the area of the Los Angeles Coastal Plain (basin), where three major problems present themselves. These are, 1) an already serious depression of the ground-water level, 2) an increasing demand for water, following predicted population trends, and 3) the encroachment of sea water into the basin. To combat this, and other threatened water shortages, the people of California voted funds of $1.75 billion last November to provide for, among other things, the transportation of water from the Feather River to Los Angeles. With this imported water injected into the basin, it is hoped that solutions will be provided for all three of the aforementioned problems.

Unfortunately, however, the water cannot be injected at random. Injection at one point might result in flooding in other areas. Thus it becomes necessary to know something of the dynamics of the basin, which in turn implies the need for some sort of computer simulation. As some of the flow parameters are not accurately known, the resulting cut-and-try aspects of the problem make the analog approach particularly attractive.

*Formerly with EAI Computation Center

Geology and Hydrology

The Los Angeles basin is bounded on the north by the Santa Monica Mountains, Elysian Hills, and Merced Hills; on the east by the Puente Hills and the Los Angeles-Orange County line; and on the south and west by the Pacific Ocean and the Palos Verdes Hills. Running diagonally across the basin from Southeast to Northwest is the Newport-Inglewood Uplift. The mountains and hills present complete barriers to the flow of water, while the Uplift presents a resistance to flow that varies along its length. The flow across the Orange County line is relatively small and is considered constant. The boundary condition along the sea-coast is met by holding the ground-water level at sea level. Finally, a constant head is maintained at the Whittier Narrows. This completes the set of boundary conditions.

The water-bearing sands and gravels, called aquifers take roughly the form of a series of overlying sheets of varying thickness. The aquifers are separated from each other in the main by clay lenses but make contact at numerous locations. The areas of contact are relatively small, but they provide the principal means for the transfer of water between aquifers.

For the initial computer analysis, the subject of the present paper, the ground-water complex described above is replaced by a very much simplified model. This model consists of a single unconfined aquifer, whose local properties are composites of the corresponding properties of the several aquifers that make up the actual structure. The thickness of the single aquifer is considered to be small compared to its lateral dimensions, and its bounding edges are irregular in shape. Finally, time varying flows are extracted from or injected into the aquifer in a distributed fashion. A plan view of the aquifer is shown in Fig. 1.

Mathematical Model

The equation of continuity of an unconfined aquifer, in which there is no vertical variation of properties, is given by

$$\nabla \cdot \rho \bar{v} + S \frac{\partial h}{\partial t} + Q = 0 \qquad (1)$$

where $h = \delta + z$, $\delta = \delta_o + \bar{\delta}$. δ_o and $\bar{\delta}$ are the mean and perturbation of δ. Neglecting gravitational forces, Darcy's Law gives the following equation of motion

$$\bar{v} = - \rho g \frac{k}{\mu} \nabla h \qquad (2)$$

The symbols incorporated in Eqs. (1) and (2) are defined as follows

z	= reference elevation	L
h	= head	L
δ	= local thickness of saturated portion of the aquifer	L
$\bar{v}$	= velocity	LT^{-1}
S	= storage coefficient	dimensionless
Q	= volumetric flow rate per unit area	LT^{-1}
ρ	= density	ML^{-3}
g	= acceleration of gravity	LT^{-2}
k	= permeability	L^2
μ	= absolute viscosity	$ML^{-1}T^{-1}$
t	= time	T

Equations (1) and (2) are combined and linearized to yield a single equation which, subject to appropriate boundary conditions, describes the dynamics of flow in the aquifer. The thickness of the aquifer is assumed small compared to its lateral dimensions. This equation is

$$\nabla \cdot T \nabla h - S \frac{\partial h}{\partial t} - Q = 0 \qquad (3)$$

where $T = \delta_o \rho g k / \mu$. The quantities T and S are, respectively, the local transmissibility and storage coefficient of the aquifer. The source flow rate, Q, is in most cases time dependent. This flow rate is the algebraic sum of several component extraction and replenishment flows. The replenishment flows are precipitation, imported water, stream percolation, artificial recharge, and subsurface inflow across boundaries. The extraction flows consist mainly of the water pumped from the aquifer for consumptive use, and subsurface outflow across boundaries.

In this work Eq. (3) is replaced by an equivalent system of difference-differential equations, the simultaneous solution of which gives the wanted function h at a finite number of node points lying within the boundaries of the aquifer. The system of equations is solved by means of a general purpose electronic differential analyzer.

If the node points at which the difference-differential equations are defined are regularly spaced, the equations have a particularly simple symmetrical form that is identical for all the node points. However, there are two troublesome problems which often exist if such spacing is to be used. The first of these comes about if the boundaries are irregularly shaped. The node points near such a boundary must be connected to the boundary by grid elements of irregular lengths. This necessitates the use of special equations at these points. The second problem concerns the change of mesh size within the boundaries. In the neighborhood of expected large rates of change of the wanted function, the mesh size must be reduced if an accuracy is to be obtained that is comparable with the accuracy in the rest of the aquifer. Since it would be uneconomical from an equipment utilization point of view to preserve a minimum spacing of node points throughout the aquifer, a rational approach to the problem of mesh size change is required.

Both of these difficulties are encountered in the case of the thin aquifer, shown in Fig. 1. The boundaries are indeed irregular and the expected spatial rates of change of the head, h, and the properties of the aquifer (i.e., local transmissibility, etc.) are large. Extensive work has been done by MacNeal(1) in the development of an asymmetric network of node points for two dimensional second order b.v. problems which overcomes the difficulties outlined above. Such a network pertaining to the thin aquifer is shown in Fig. 1. The attendant system of difference-differential equations is

$$\sum_i (h_i - h_B) Y_{i,B} = A_B S_B \frac{dh_B}{dt} + A_B Q_B$$
$$Y_{i,B} = \frac{J_{i,B} T_{i,B}}{L_{i,B}} \qquad (4)$$

where

A_B	= area associated with node B	acres
$Y_{i,B}$	= conductance of path between nodes i and B	$\frac{\text{acre ft}}{\text{year ft}}$
S_B	= storage coefficient of polygonal region associated with node B	dimensionless
$Q_{i,B}$	= volumetric flow rate per unit area at node B	$\frac{\text{acre ft}}{\text{year acre}}$
$T_{i,B}$	= transmissibility at midpoint between nodes i and B	$\frac{\text{acre ft}}{\text{year ft}}$
$L_{i,B}$	= distance between nodes i and B	ft
$J_{i,B}$	= length of perpendicular bisector associated with nodes i and B	ft

A typical node point, its neighbors, and the polygonal region associated with it is shown in Fig. 3.

The terms in the first of Eqs. (4) are interpreted physically with the aid of Fig. 3 as follows. The left hand side represents the sum of the flows within the aquifer to the polygonal region, A_B, (across the dashed lines, whose lengths are $J_{i,B}$). The first term on the right hand side represents the rate of water storage within A_B.

The remaining term represents the extraction or replenishment flow from A_B.

Computer Circuits

The system of equations (4) can be represented by a network of resistors, capacitors, current generators, and arbitrary function generators. The portion of such a network that corresponds to the node point array of Fig. 3 is shown in Fig. 4.

Special purpose computers containing multi-decade resistor and capacitor elements, current generators and arbitrary function generators, have been constructed in the past to solve diffusion problems. These computers are sometimes referred to as network analyzers. The current generators in such computers are often formed from two operational amplifiers.

With the exception of the multi-decade resistors and capacitors, all of the above equipment is normally found in a general purpose electronic differential analyzer. In place of the multi-decade passive elements, the differential analyzer contains a large quantity of single high precision passive elements. By utilizing this equipment appropriately, the differential analyzer can be made to simulate the network analyzer with no substantial overall increase in equipment utilized per node. Thus, in the present case, the general purpose analog computer has the capability of economically simulating the network analyzer, as well as the ability to solve other classes of problems.

The system of equations (4) can be mechanized in the usual way as shown in Fig. 5. However, this approach has two unattractive features. First, the conductance $Y_{i,B}$ is represented by two different pot settings for every node. A slight error in setting one of the two has the effect of introducing an additional flux term. For example, if the feedback pot is set slightly high, one can think of the resultant error as a leakage flux. This can sometimes be disproportionately large compared to the error in pot setting, especially in cases where the flux should be zero in steady-state.

A second undesirable feature of the circuit is the computational labor involved in a change in conductance value, and that at least two pots must be reset. This difficulty is made particularly unattractive by the change in conductance values required by the model tailoring.

To avoid these difficulties, and at the same time retain an economy of equipment, two non-standard circuits are used; a voltage divider circuit and a variable capacitor circuit. These are shown in the final diagram of Fig. 7.

Voltage Divider Circuit

Each of the terms $(h_i-h_B)Y_{i,B}$ of Eq. (4) is mechanized by a simple voltage divider potentiometer network between the two nodes. Pairs of 60,000 ohm resistors, matched to within a percent, are chosen for compatibility with the 30,000 ohm pots. By setting the conductance pots with one node at reference and the other at ground, loading of the dividers is compensated for. The outputs of these pots now read directly the flow between two adjacent nodes.

Variable Capacitance Circuit

To obtain a variable capacitance, a pot can be inserted in series with the integrating capacitor. The computer modification is minor, and allows one switch to restore the integrator to its normal mode. The circuit of Fig. 6 represents a typical application of this technique, and has an exact transfer function of:

$$\frac{e_o}{e_i}(s) = - \frac{\frac{1}{R_1 Cs}}{\frac{1}{\mu} + \frac{\delta\left[1-Cs(1-\delta)R/\mu\right]}{1+\delta(1-\delta)RCs} + \frac{1}{\mu R_1 Cs} + \frac{1}{R_f Cs} + \frac{1}{\mu R_f Cs}}$$

If $\mu > 10^4$; $R, R_1, R_f > 10^4$; $C < 10^{-6}$

$$\frac{e_o}{e_i}(s) = \frac{\frac{1}{R_1 Cs}}{\frac{\delta}{1+\delta(1-\delta)RCs} + \frac{1}{R_f Cs}}$$

Or, if $\tau \equiv \delta R_f C$

$$\epsilon \equiv \frac{R}{R_f}(1-\delta)$$

$$\frac{e_o}{e_i}(s) \cong -\frac{R_f}{R_1}\,\frac{1+\epsilon\tau s}{1+(1+\epsilon)\tau s}$$

The transfer function of the integrator alone ($R_f = \infty$) is

$$\frac{e_o}{e_i}(s) = -\frac{1}{\delta R_1 Cs} - \frac{R}{R_1}(1-\delta)$$

With $R = 3 \times 10^4$, $R_1 = R_f = 10^6$, and $C = 10^{-6}$, the step response of this circuit can have a maximum initial error of 3% of steady-state, dropping to .02% in one time constant. With capacitors of 10^{-8}, as might be used in the repetitive mode of operation, these errors become entirely negligible.

Computer Model Check Procedures

The number and similarity of the analog circuits in this simulation suggest some special checking procedures. These are outlined in the order in which they are best carried out.

Uniform Head Distribution

If all boundary values and source flow

rate functions are removed from the aquifer, the head distribution corresponding to a reference voltage applied at any one node should be uniform at steady-state. In order to produce this result it is necessary that there be no regenerative loops (i.e., the inverting amplifiers be properly placed).

Single Degree of Freedom Dynamic Check

The dynamic behavior of the portion of the aquifer associated with one node, under the influence of the source flow rate function, is checked against a precalculated function. To implement this check, all nodes are held at ground potentional except the node of interest, and the computer allowed to integrate. A typical source flow rate function and the dynamic response are shown in Figs. 8 and 9 respectively. This test is made at each interior node, and completely checks the individual nodes.

Model Flux Balance

As a final analog circuit check, an overall nodal flux balance is made. With all boundary conditions set in, and constant flow rates inserted, the aquifer is allowed to reach equilibrium. At each node, the algebraic sum of the flows should equal the source flow rate. Moreover, a balance can be made of the total water entering the aquifer against the total water leaving. In addition to its usefulness as a check, the nodal balance can provide the hydrologist with some insight as to the hydraulic behavior of the aquifer. This can be of considerable aid in the adjustment of the flow parameters in the model tailoring phase of the problem.

Model Tailoring

In many areas of the aquifer, the transmissibilities are difficult to calculate from available data. Consequently, a "best value" is used initially. A complete set of plots is then generated by the computer and compared with historical data. Where discrepancies occur, conductance pots are adjusted and new plots made. As this portion of the program can be quite time consuming, it points toward the high desirability of the repetitive mode of operation. While we did not have this feature when performing this study, we do plan to use it next time.

Data Gathering

After the model is perfected to the hydrologists' satisfaction, a study is usually made of the dynamic behavior of the basin. The present model is linear. Hence, an exhaustive study of the model could be made by obtaining influence functions. That is to say, one could plot the response (head vs time) of the entire network due to the sudden application of a unit flow rate at any node (all other source flow rates are set to zero). Similar responses would be obtained for successive applications of the unit flow rate at all other interior nodes. Any desired solutions could be constructed from these influence functions by superposition. However, this procedure normally results in a very large number of curves. Hence, it is usual to make investigations of the dynamic behavior of the aquifer under certain specific conditions of replenishment or extraction that are expected to take place in the future. For example, in the present study, sets of responses were obtained for large constant injection flow rates in the Whittier Narrows area, the Los Angeles River area, and the Manhattan Beach area, etc. Similar data was taken for large extractions in the Lakewood area that would be expected from future industry.

Acknowledgements are in order to Lloyd Fowler, Robert Thomas, and Robert Chun of the California State Department of Water Resources, Southern California Office; and to Harry Turner of Electronic Associates for suggestions leading to the electronic differential analyzer circuits used in this work.

1. MacNeal, R.H., "An Asymmetrical Finite Difference Network", Quarterly of Applied Mathematics, Vol. XI, No. 3, 1953.

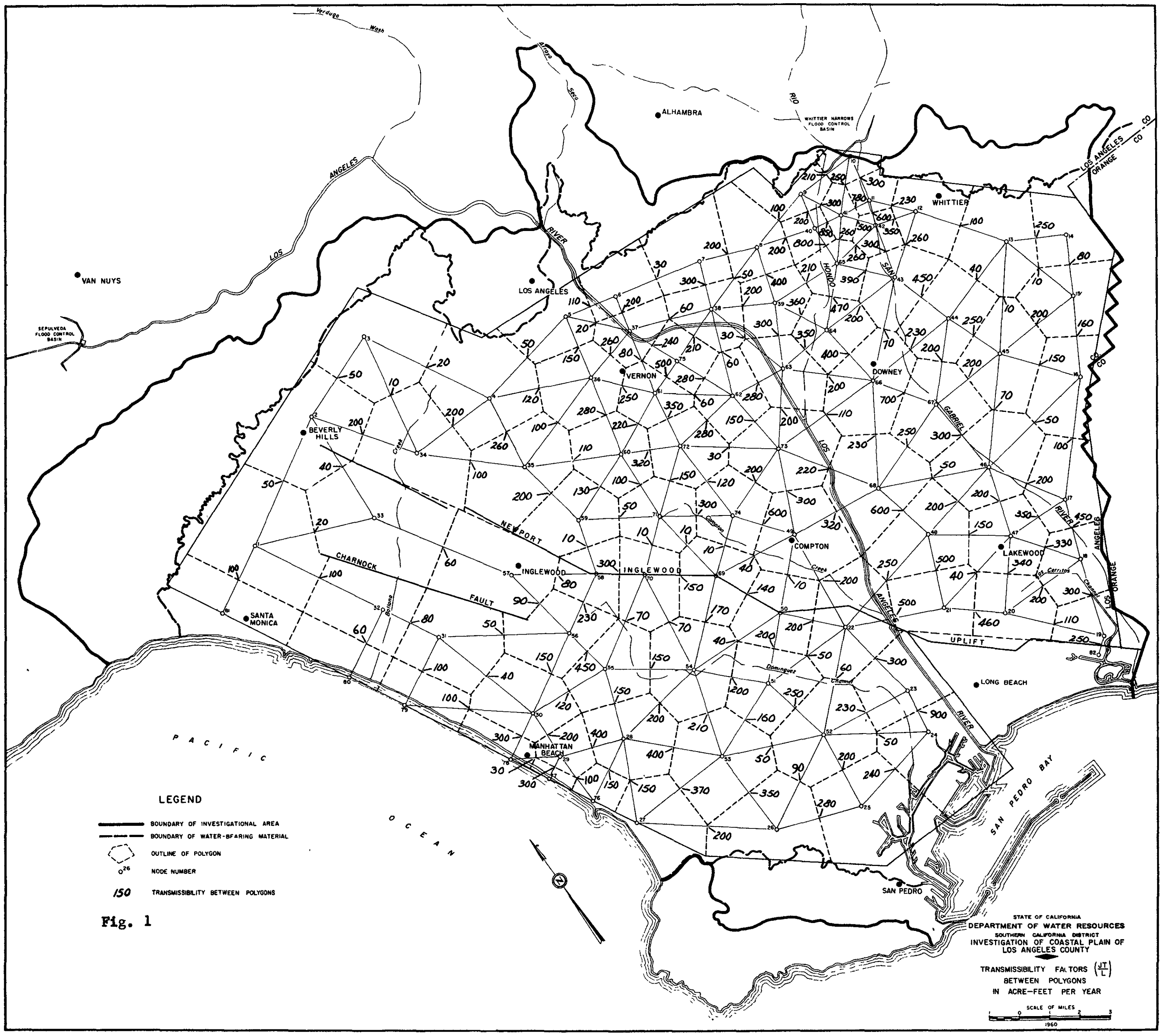
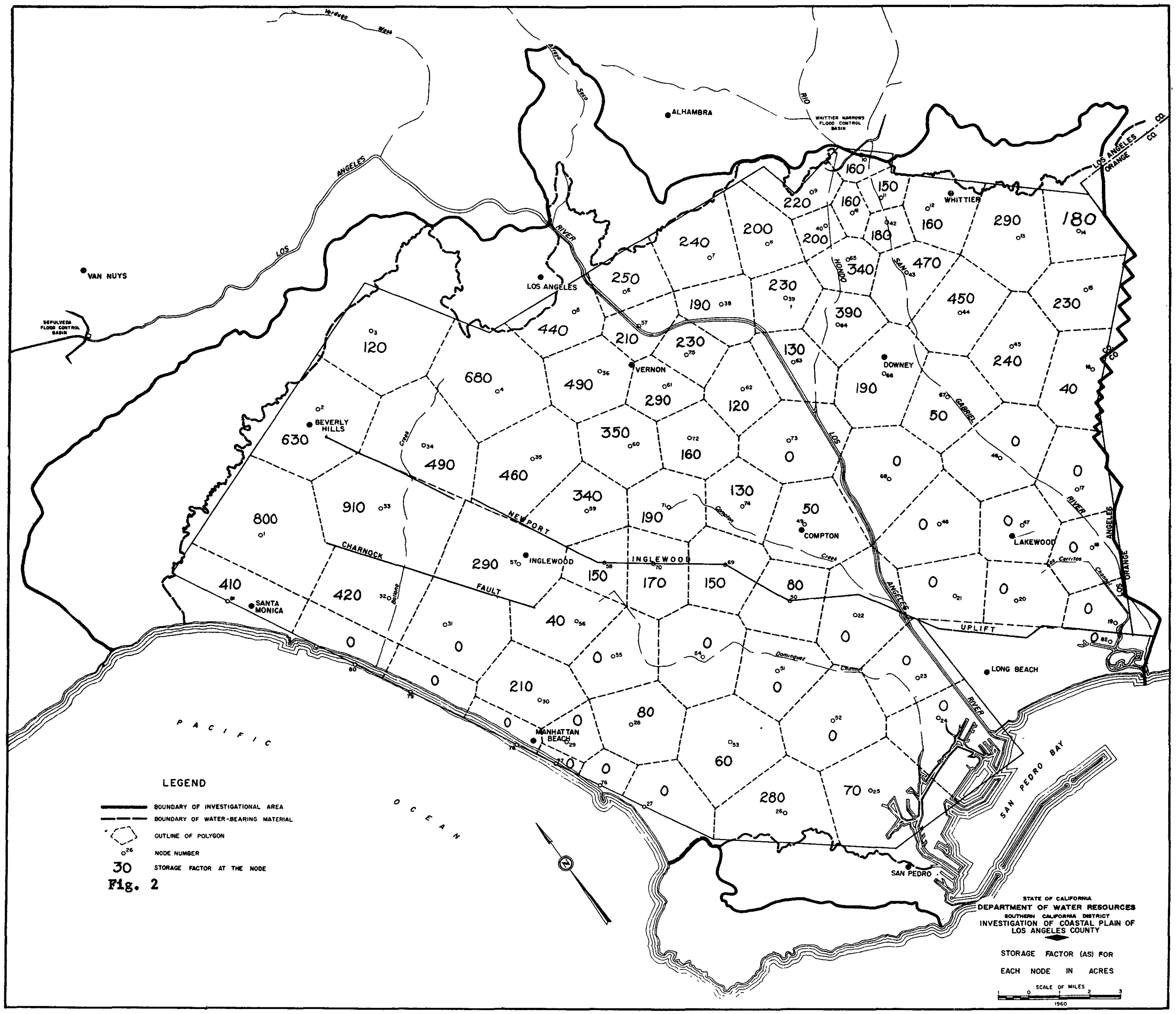

LEGEND
BOUNDARY OF INVESTIGATIONAL AREA
BOUNDARY OF WATER-BEARING MATERIAL
OUTLINE OF POLYGON
NODE NUMBER
150 TRANSMISSIBILITY BETWEEN POLYGONS
STATE OF CALIFORNIA
DEPARTMENT OF WATER RESOURCES
SOUTHERN CALIFORNIA DISTRICT
INVESTIGATION OF COASTAL PLAIN OF LOS ANGELES COUNTY
TRANSMISSIBILITY FACTORS (JT/L) BETWEEN POLYGONS IN ACRE-FEET PER YEAR
SCALE OF MILES
1960
VAN NUYS
SEPULVEDA FLOOD CONTROL BASIN
LOS ANGELES RIVER
Verdugo Wash
Arroyo Seco
ALHAMBRA
RIO HONDO
WHITTIER NARROWS FLOOD CONTROL BASIN
WHITTIER
LOS ANGELES CO
ORANGE CO
LOS ANGELES
BEVERLY HILLS
VERNON
DOWNEY
SAN GABRIEL RIVER
NEWPORT INGLEWOOD UPLIFT
CHARNOCK FAULT
INGLEWOOD
COMPTON
LAKEWOOD
SANTA MONICA
Ballona Creek
Compton Creek
Coyote Creek
Dominguez Channel
Cerritos Channel
LONG BEACH
MANHATTAN BEACH
SAN PEDRO
SAN PEDRO BAY
PACIFIC OCEAN

Fig. 1


LEGEND
BOUNDARY OF INVESTIGATIONAL AREA
BOUNDARY OF WATER-BEARING MATERIAL
OUTLINE OF POLYGON
NODE NUMBER
30 STORAGE FACTOR AT THE NODE
STATE OF CALIFORNIA
DEPARTMENT OF WATER RESOURCES
SOUTHERN CALIFORNIA DISTRICT
INVESTIGATION OF COASTAL PLAIN OF LOS ANGELES COUNTY
STORAGE FACTOR (AS) FOR EACH NODE IN ACRES
SCALE OF MILES
1960
PACIFIC OCEAN
SAN PEDRO BAY
VAN NUYS
ALHAMBRA
WHITTIER NARROWS FLOOD CONTROL BASIN
SEPULVEDA FLOOD CONTROL BASIN
LOS ANGELES
BEVERLY HILLS
VERNON
DOWNEY
WHITTIER
SANTA MONICA
INGLEWOOD
COMPTON
LAKEWOOD
LONG BEACH
MANHATTAN BEACH
SAN PEDRO
NEWPORT INGLEWOOD UPLIFT
CHARNOCK FAULT
LOS ANGELES RIVER
SAN GABRIEL RIVER
RIO HONDO
LOS ANGELES CO.
ORANGE CO.
Fig. 2

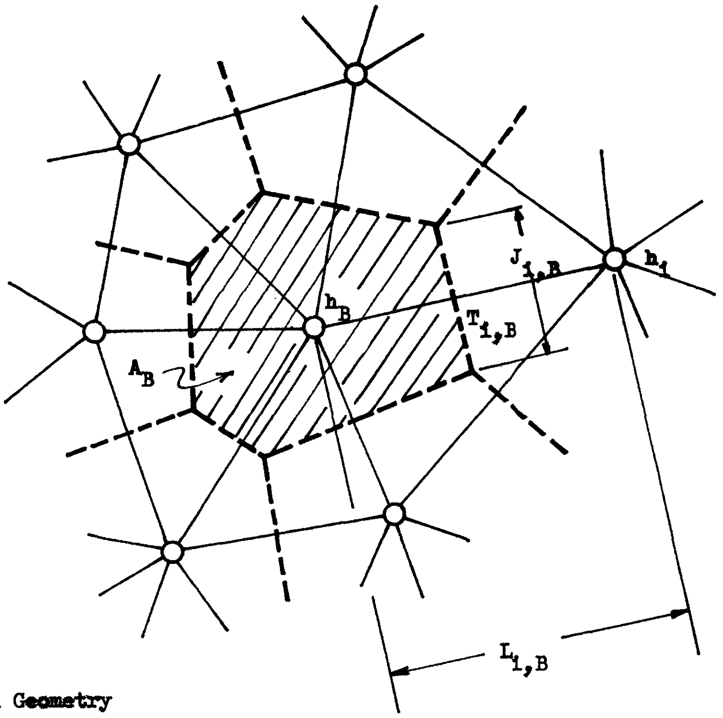
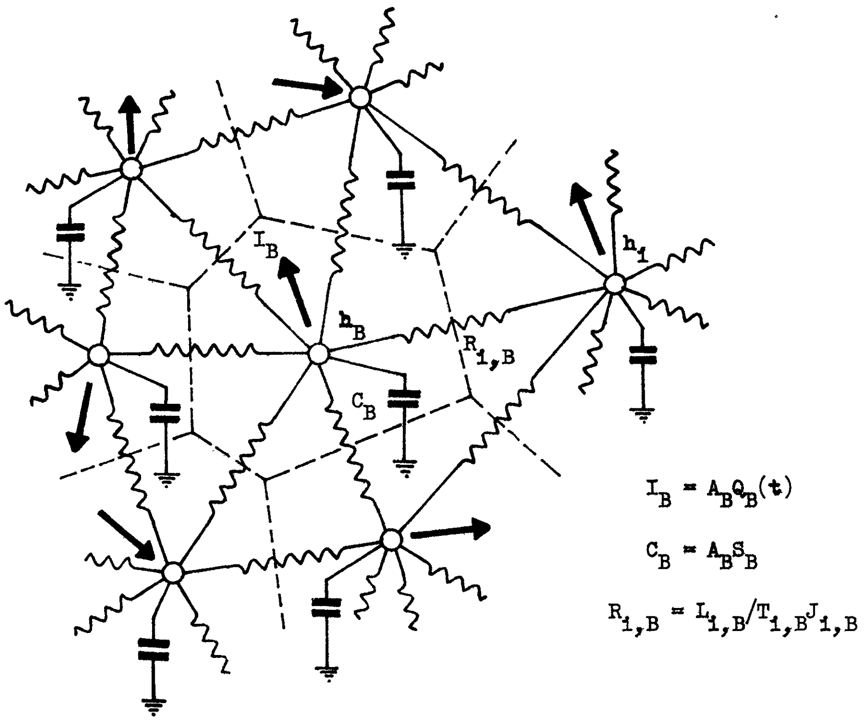


Fig. 3. Polygon Geometry



$I_B = A_B Q_B(t)$

$C_B = A_B S_B$

$R_{i,B} = L_{i,B}/T_{i,B}J_{i,B}$

Fig. 4. Typical R-C Network

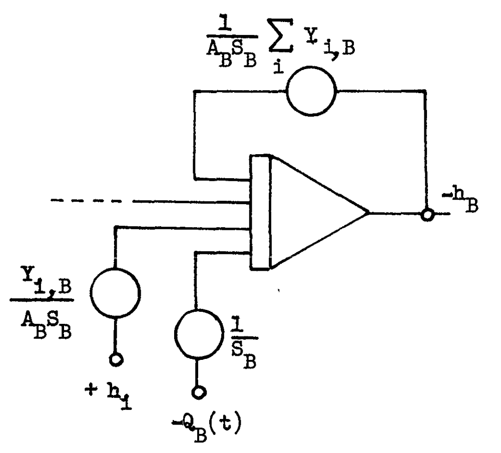


Fig. 5.

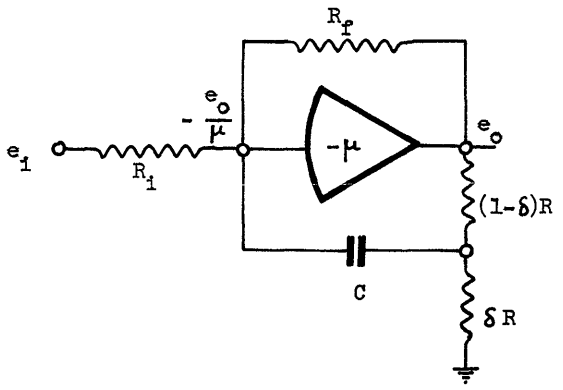


Fig. 6.

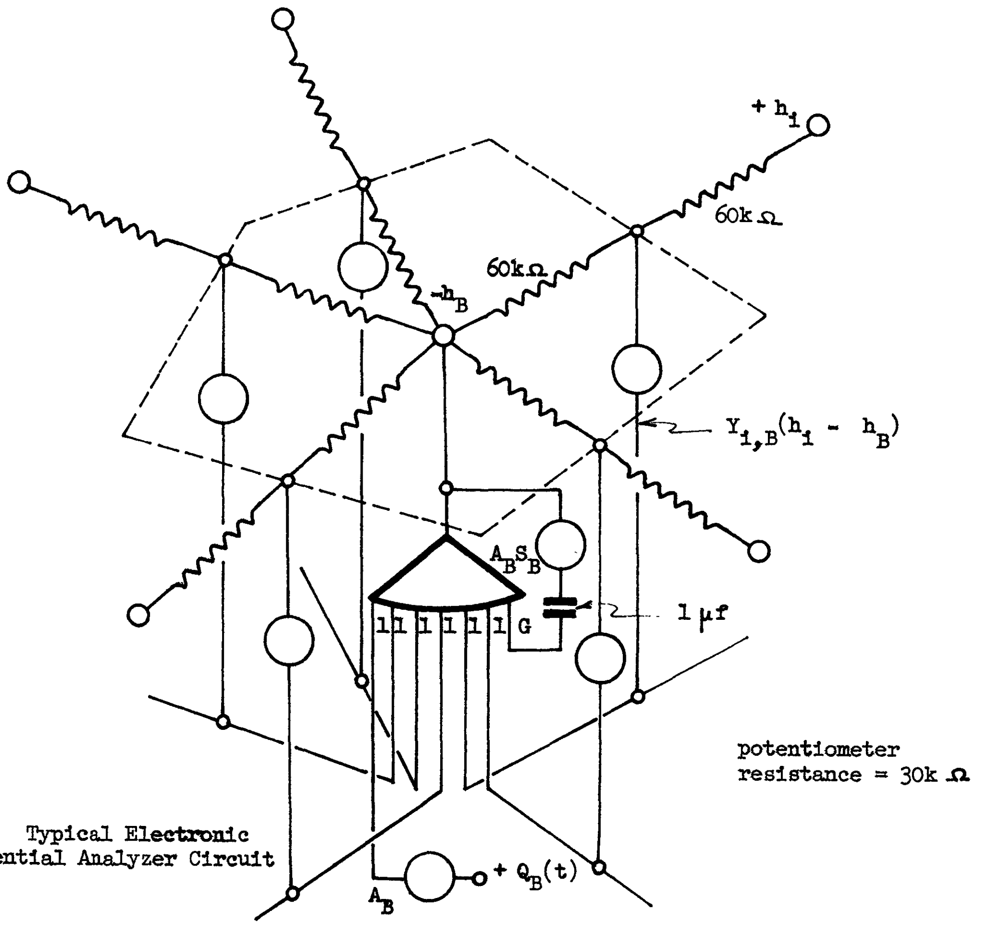


Fig. 7. Typical Electronic Differential Analyzer Circuit

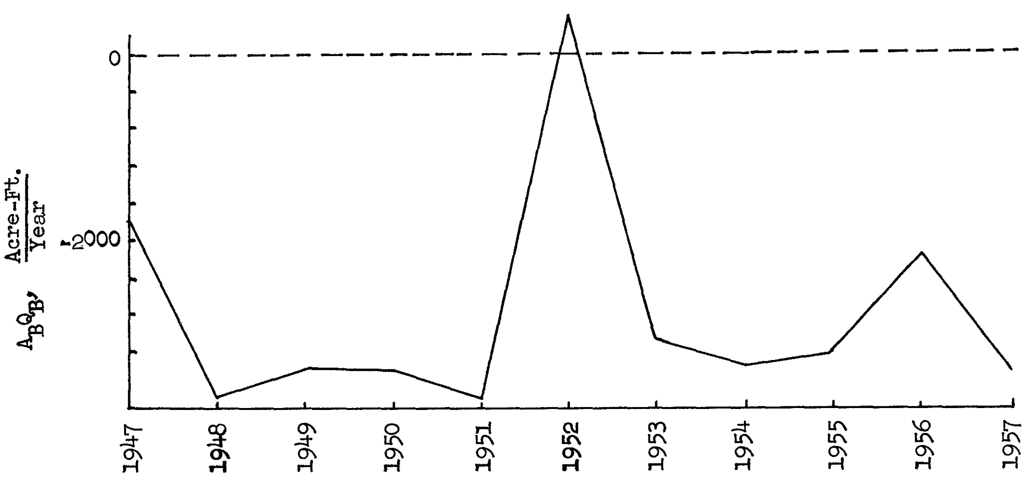


Fig. 8. Source Flow Rate Function

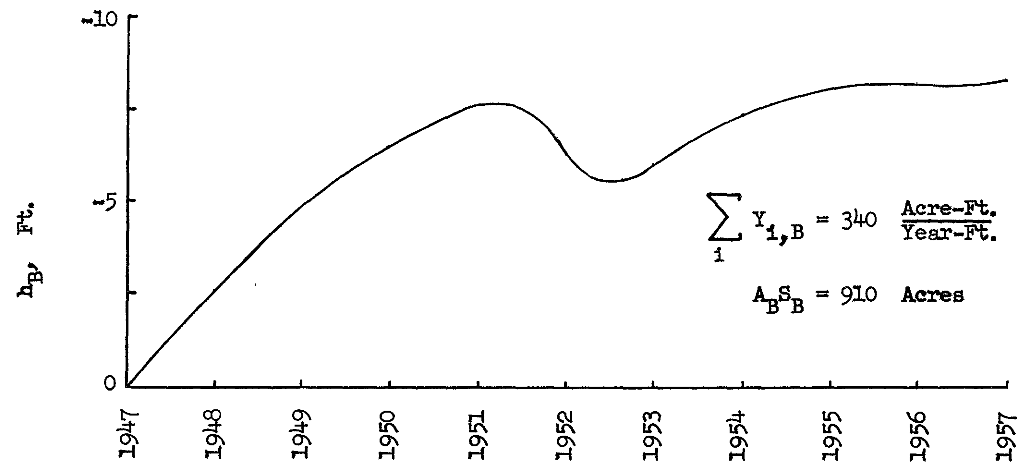


Fig. 9. Dynamic Response to Source Flow Rate Function

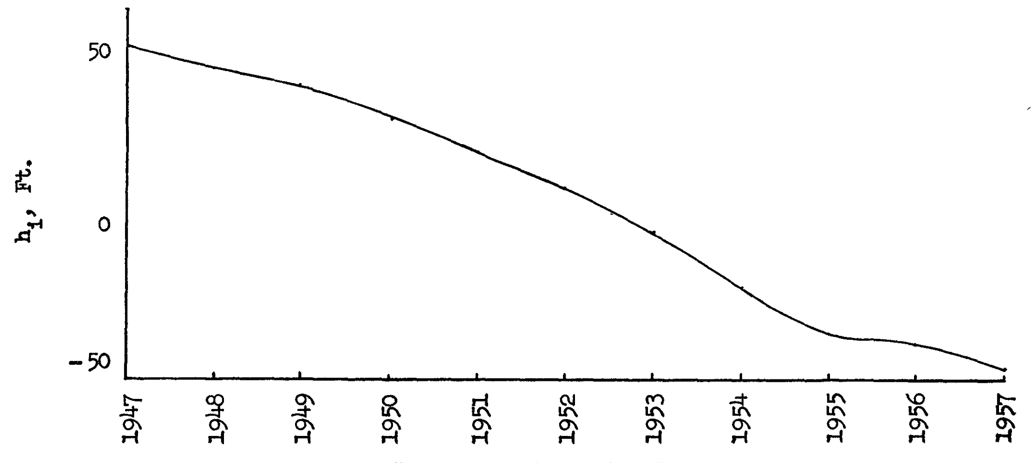


Fig. 10. Typical Local Hydrograph

A SELF ORGANIZING RECOGNITION SYSTEM

J. R. Singer
The Miller Institute for Basic Research in Science
and the Electronics Research Laboratory
University of California
Berkeley, California

INTRODUCTION

The design of a system for recognition of patterns or objects with provision for size and rotation invariance[1,2,3] will be described. This system is also self-learning in that a program of remembering a pattern by "seeing" that pattern is incorporated in the system. The process of "free learning" which is defined here as abstracting from the results of numerous inputs and outputs to arrive at a more general recognition class is also part of our design. We shall first briefly describe the properties incorporated in our design, and then outline the system.

The nature of recognition is such that the image of each object belongs almost to an infinite class in the sense that a retinal or photoreceptor matrix array can be stimulated in a very large number of different modes by **a** single object, dependent upon both the distance from the object to the retina and the rotation of the object. Therefore the first design task is the recognition of the <u>same</u> object with size and rotation invariance. The second important design principle is that there be a recognition of a set of objects which are the same in the sense that a single specific output is required for all members of the set even though the members all vary somewhat image-wise (e.g., all A's are members of one set even though not drawn in the same way). Third, the coded differences between different sets of images (A's, B's, C's, etc.) must be maximized in order to also maximize the permissible variation <u>within</u> a set. These three guidelines for design are to be incorporated in the system logic so as to permit the recognition procedure to interpolate internally by a type of self-learning process.

No learning can take place without a feedback or correcting signal. When humans learn to read, they are given a simultaneous set of inputs, i.e., they are shown letters and asked for a specific response to those letters. After a time, the human learns to internally program his learning. That is, by trying out various responses . to stimuli the desired response is reached using a nebulous error correcting signal. We can design in a much simplified way towards an analogous electronic system. Initially, the system learns patterns by storing the images in the memory when the machine is in a learning mode. Later, when the machine is in a reading (or recognition) mode, the input images are compared with the memory patterns and a <u>measure</u> of the "fit" is obtained. By internal programming, the machine accepts and recognizes the <u>worst</u> fit for a set of images which does not overlap with a different set of images. Later on, by internal programming, the internal system re-examines the recognition decisions to "learn" to make better ones.

MATHEMATICAL DESCRIPTION OF AN IMAGE DILATION

A fundamental requirement for electronic recognition of objects independent of the image dimensions is a transformation to a standard image representation. The image need not be optical; sound falls into the same category. The absolute amplitude of signal is not of paramount importance in general, but the relative amplitude or size of each signal element to the rest of the signal is an essential component for size-invariant recognition. We wish to discuss first the mathematical analysis of dilations and then show that such transformations may be carried out using electronic techniques.

The purpose of this exposition is to show that a practicable recognition system can incorporate the important property of recognition regardless of size by means of a relatively simple electronic system. Such a system has been outlined previously by the author in a less detailed manner[1,2] without a rigorous formulation. This paper will formalize the approach and result in a more mathematically satisfying exposition.

We consider, first, the dilation transformation as a homothetic transformation having a unique ordinary point at its center[4]. For rectangular coordinates the transformation is simply

$$x' = ax + c_1 \qquad y' = ay + c_2 \tag{1}$$

where a is the dilation scale factor and is not zero or one. The transformation has a center given by

$$x = c_1 (1 - a)^{-1} \qquad y = c_2 (1 - a)^{-1} \tag{2}$$

The set of dilation transformations does not form a group since it is not closed under multiplication, nor is there an identity transformation.

If we consider an image of a figure on a plane with border coordinates x_i y_i with the centroid of the image initially centered, the constants c_1 and c_2 may be set to zero in the transformation. The unique point of the dilation is then (0,0) which is the center of the image.

The dilation of an image occurs externally to the photoreceptor system, and may be considered as a projection of an image as the projector is brought closer to the receptors. Similarly, of course, an image size reduction may be interpreted as a projector withdrawal from the receptors. The physical problem consists of arranging the image reception system so as to provide the same digitized image code for the image regardless of dilation (except that the image must be within the receptive area of the photoreceptors). A simpler dilation transformation than (1) may be utilized if we work in polar coordinates. The corresponding polar dilation is

$$\begin{aligned} r' &= ar \\ \theta' &= \theta \end{aligned} \qquad (3)$$

Therefore the simplest arrangement of photocells is a polar array. The constant of dilation (a) is not constant for all projected images, but takes all possible values up to the maximum value

$$a_{max}\, r_o = r_{max} \qquad (4)$$

where r_o is the minimum radius of a resolved image centered on a polar arrangement of photocells, and r_{max} is the maximum possible radius for a resolved image contained in the photocell array.

PHYSICAL DESIGN

A physical embodiment of the general principles is shown in Figure 1. The photocells are arranged in a polar array to create an electronic "retina" which provides for a set of pulses which can be logically connected so as to provide a size-invariant image code. Each segment in a sector of the polar arrangement of photoreceptors contains four or more independent photocells which are connected to provide an output only if an image <u>border</u> is projected upon a segment. The method of combining photocells four at a time is shown in Figure 2. The logical (Boolean) analysis of requiring one, two, or three photodetectors to be in darkness (or illuminated) but not all four to be equally stimulated to obtain an output, is

$$(a+b+c+d) \cdot (\overline{a \cdot b \cdot c \cdot d}) \qquad (5)$$

where a, b, c, d represent the binary state of each cell, a bar represents "not", the dot is for logical "and", and the + is for logical "or".

Using this model of a bridge circuit, each photoreceptive border receiver area must become larger with increased distance from the center of the polar array of Figure 1. There are several methods of accomplishing the increased reception area. If solid state photoreceptors are utilized, the required area is easily arranged. However, another approach utilizes glass (also quartz, etc.) fibers to guide the light paths to the photocells. This permits a very small area to be resolved. It is particularly useful to have very small photoreceptor areas in towards the center of the polar array to decrease the area of the central blind spot, and fiber light guides may be invaluable for application here. There is, however, a more important reason for utilizing fiber optical wave guides to collect light and direct it to the photoreceptors. The reason is economy of electronic elements with essentially no loss of resolution. Away from the center of the polar array of photodetectors, the segments are large. Only one pulse output signal must come from each segment. In order to "see" fine points in the outer segments, individual photosensitive elements must be smaller than the point. By using thin glass fibers leading more or less randomly (within a segment) to four (or eight) bridge-connected electronic photoreceptors, the resolution of a small signal (narrow line) is good, while the number of electronic components is relatively small. The effect of this arrangement is to permit even one stimulation of a single light fiber to cause a signal to emerge from one polar segment. The more or less random connections of the optic fibers to the photocell bridge lead to a possibility of cancelling a set of light spots which stimulate the four photodetectors of the bridge equally. Due to the lack of well-organized connections, the probability of such a pattern occurring is almost nil.

It may be interesting to note that the human retina may be organized in an analogous way. Near the center of the retina only a few photosensitive retinal elements (rods) are interconnected neurally before being connected to a fiber of the optic nerve. The rods (or cones) towards the periphery of the retina are interconnected sometimes hundreds at a time before being connected with one optic nerve fiber[5].

We have discussed the arrangement of the photocells into bridges of fours with each set of four photoreceptors filling a segment of the polar plot of Figure 1. The next topic to be considered is the logical arrangement of output signals from the polar array. The output is most readily and conveniently manipulated by a general purpose digital computer. Even though many of the computer operations will not be utilized, the availability of such computers is an economical gain.

THE PHOTOMATRIX

The photodetectors are energized by a pulsed voltage source for a short time -- the order of a microsecond -- just long enough to exceed the response time of the light detectors. The repetition rate for this activating pulse is relatively slow (about 30,000 pulses per second). The repetition rate of $1/r_t$ of the response time, where r_t is the total number of circular divisions of photoreceptor areas, is utilized to code the image for recognition or memorization prior to the appearance of a second image.

The outputs from the polar segments are connected as shown in Figure 3. Each output goes through a unilateral conduction element (a diode) and a delay (of about a microsecond which is about the receptor response time) before being attached to radially neighboring photoreceptors. One has a choice of directing the output pulse path either radially outward or inward, and we have elected, for the present system, to arbitrarily use radially outward propagating signal pulses.

Let us now observe the effect of our logical and geometrical circuit combination. Suppose an image appears on the photoreceptor array shown in Figure 1. The borders of the image which fall upon segments of the array activate the photoreceptor bridges (Figure 2) lying in those segments (when a power pulse occurs) and a set of pulses representing the borders travels outwards along the radial connecting lines shown in Figure 3. The activating pulse also starts a "clock" which determines the position of all pulses on a time scale. We have, then, thirty-six radial lines either carrying pulses or not with the pulse position (in time) indicating the borders of an image. We may consider the digitized image representation to be a matrix of binary elements with the rows corresponding to the radial output lines and the columns corresponding to the time scale (each column is separated by one delay period or clock time unit).

THE MATRIX REPRESENTATION OF AN IMAGE

The image is converted to a set of radially outward traveling pulses by the reception system described above. We now wish to describe the combination of pulses to represent the image for both learning and recognition purposes.

It is convenient to number the radial pulse propagating lines in a clockwise manner. The most vertically upward sector then of Figure 1 is called radial number 36 and the sectors to the right are radials 1, 2, 3, etc. If the radials are given rows in a matrix of binary numbers then the time sequence of pulses in each radial determines the position of the units in the matrix. The matrix size is given by: the number of radial lines = number of rows of the matrix, and the number of delay units = the number of columns of the matrix. We assume here that each delay unit is equal between segments of the photodetector array and that a unit delay follows the outermost segment.

Having described the image coding scheme, we will now illustrate how images which are centered upon the polar photoarray provide a binary matrix code. Consider images of alpha-numeric (letters and numbers) form first. Suppose that the width of the lines of these letters are larger than the diameter of a photoreceptor element (e.g., a glass fiber conductor), and also that the image is of a size suitable for total inclusion in the photosensitive array.

For heuristic simplicity consider an "0" first. The figure is imaged, the photocells energized by a pulse, and the pulses representing the "0" travel down the radial delay lines in synchronism with the digital clock. In conformity with common practice, absence of a pulse denotes a zero, and a pulse denotes unity. The matrix denoting the "0" code is then obtained by taking the number one radial "word" (set of pulses) as the first row of the matrix followed by the number two radial word, etc. The form of the matrix is indicated in abbreviated form as

$$\begin{bmatrix} 0\ 0 \ldots 1\ 0 \ldots 0\ 1\ 0\ 0 \ldots 0 \\ 0\ 0 \ldots 1\ 0 \ldots 0\ 1\ 0\ 0 \ldots 0 \\ 0\ 0 \ldots 1\ 0 \ldots 0\ 1\ 0\ 0 \ldots 0 \\ 0\ 0 \ldots 1\ 0 \ldots 0\ 1\ 0\ 0 \ldots 0 \\ 0\ 0 \ldots 1\ 0 \ldots 0\ 1\ 0\ 0 \ldots 0 \\ 0\ 0 \ldots 1\ 0 \ldots 0\ 1\ 0\ 0 \ldots 0 \\ \cdot \quad \cdot \qquad \cdot \qquad \cdot\ \cdot \\ \cdot \quad \cdot \qquad \cdot \qquad \cdot\ \cdot \\ \cdot \quad \cdot \qquad \cdot \qquad \cdot\ \cdot \\ \cdot \quad \cdot \qquad \cdot \qquad \cdot\ \cdot \end{bmatrix} \qquad (M1)$$

It may be noted that the zeros both to the left and right of the columns of ones provide information about the size of the image, but nothing regarding shape. Therefore we use a floating decimal point or similar type operation to eliminate all columns of zeros to the left and the right of the first columns from the left and from the right which contain units. We may retain information regarding the number of columns dropped as an indication of image size if we chose, but for the moment, that is not important. The reduced matrix is then

$$\begin{bmatrix} 1\ 0 \ldots 0\ 1 \\ 1\ 0 \ldots 0\ 1 \\ 1\ 0 \ldots 0\ 1 \\ 1\ 0 \ldots 0\ 1 \\ 1\ 0 \ldots 0\ 1 \\ 1\ 0 \ldots 0\ 1 \\ \cdot\ \cdot \quad \cdot\ \cdot \\ \cdot\ \cdot \quad \cdot\ \cdot \\ \cdot\ \cdot \quad \cdot\ \cdot \\ \cdot\ \cdot \quad \cdot\ \cdot \end{bmatrix} \qquad (M2)$$

Now, the columns containing entirely zeros do not provide much information. These only indicate the "thickness" of the image. Nonetheless, the thickness is an invariant of an image; due to the photoreceptor geometry, a dilation of the image provides the same thickness in the matrix representation. Therefore, we choose not to operate upon the "internal" columns containing all zeros. The reduced matrix M2 is called the "canonical matrix" for "0".

We next consider the redundancy of information contained in matrix M2. It is apparent from a cursory examination of the matrix that the outer shape (outer border) of a figure is represented primarily by the change of position of the occurrence of units (i.e., nonzero elements) in the right hand columns. Therefore, it is reasonable to utilize the difference between row positions of the right hand units in the canonical matrix. For the M2 matrix the difference is given by a string of zeros since all of the units occupy the same row. The canonical outer border is then:

0, 0, 0, 0, 0, . . . , 0 (C1)
(36 zeros)

This word is stored in the memory as a representation of an O or zero when the system is in the learning mode.

It will be useful to next give an example of a less symmetrical pattern such as a line drawing of the letter D. This letter is shown "centered" in the photomatrix of Figure 4. The canonical matrix of the letter is shown in (M3).

```
                                       Sector No.
. . . 0|0 0 0 1 1 0|0 0 0 0 0          (1)
. . . 0|0 0 1 1 0 0|0 0 0 0 0
. . . 0|0 0 1 0 0 0|0 0 0 0 0
. . . 0|0 1 1 0 0 0|0 0 0 0 0
. . . 0|0 1 0 0 0 0|0 0 0 0 0
. . . 0|1 1 0 0 0 0|0 0 0 0 0
. . . 0|1 0 0 0 0 0|0 0 0 0 0
. . . 0|1 0 0 0 0 0|0 0 0 0 0
. . . 0|1 0 0 0 0 0|0 0 0 0 0
. . . 0|1 0 0 0 0 0|0 0 0 0 0          (10)
. . . 0|1 0 0 0 0 0|0 0 0 0 0
. . . 0|1 0 0 0 0 0|0 0 0 0 0
. . . 0|1 1 0 0 0 0|0 0 0 0 0
. . . 0|0 1 0 0 0 0|0 0 0 0 0
. . . 0|0 1 1 0 0 0|0 0 0 0 0
. . . 0|0 0 1 0 0 0|0 0 0 0 0
. . . 0|0 0 1 1 0 0|0 0 0 0 0
. . . 0|0 0 0 1 1 0|0 0 0 0 0
. . . 0|0 0 0 0 1 0|0 0 0 0 0
. . . 0|0 0 0 0 0 1|0 0 0 0 0          (20)
. . . 0|0 0 0 0 1 1|0 0 0 0 0
. . . 0|0 0 0 1 1 0|0 0 0 0 0
. . . 0|0 0 1 0 0 0|0 0 0 0 0
. . . 0|0 1 0 0 0 0|0 0 0 0 0
. . . 0|1 1 0 0 0 0|0 0 0 0 0
. . . 0|1 0 0 0 0 0|0 0 0 0 0
. . . 0|1 0 0 0 0 0|0 0 0 0 0
. . . 0|1 0 0 0 0 0|0 0 0 0 0
. . . 0|1 1 0 0 0 0|0 0 0 0 0
. . . 0|0 1 0 0 0 0|0 0 0 0 0          (30)
. . . 0|0 0 1 0 0 0|0 0 0 0 0
. . . 0|0 0 0 1 1 0|0 0 0 0 0
. . . 0|0 0 0 0 1 0|0 0 0 0 0
. . . 0|0 0 0 0 1 1|0 0 0 0 0
. . . 0|0 0 0 0 0 1|0 0 0 0 0
. . . 0|0 0 0 0 1 0|0 0 0 0 0
```

(M3). The canonical matrix representation of the letter D as imaged in Figure 4.

The canonical word representation of the letter D found by taking the difference between row positions of outer units is:

-1,-1,-0,-1,0-1,0,0,0,0,0,+1,0+1,0+1,+1,0,+1
0,-1,-2,-1,0,-1,0,0,+1,0,+1,+2,0,+1,0,-1

This is the canonical word which is stored as a memory representation of the letter D shown in Fig. 4. Of course, the digital manipulation to obtain this word is simply programmed into the machine logic and one is never aware of this representation. Once a "D" has been exposed to the photomatrix of the machine while in the learning mode, the canonical word is automatically calculated and stored in the memory.

READING THE INPUT IMAGES

Suppose that after a number of such words are memorized, the machine is in the reading (or recognition) mode. Images of letters (or words) presented to the photomatrix array are converted to canonical words which are then compared to the memorized words. The comparison may be performed in a number of ways, for example by subtraction, by sieving, or by some other logical operation. For simplicity of the heuristic design, we will assume here that the comparison is by subtraction between the memorized words and the word to be recognized. The system looks for the absolute value of the difference between all of the stored words in a certain subclass of the entire collection of the memory. (This subclass is determined by examination of canonical words for similar symmetry properties.) The initial search is for a difference of perhaps ten units or so. This would be a poor match. Suppose that three such matches at three memory addresses are found. The question now arises as to whether these three matches are members of the same class of symbols. That is, are they all memorized D's with various distortions? This question is answered by a logical procedure of comparing output signals from the three addresses. Suppose they are not identical. Next a comparison is run among the four words again, allowing for perhaps an absolute difference of five units in this comparison. Suppose that there are now two matches. A third run allowing for an absolute difference of three units or so will usually narrow down the match to one. The machine may now print out the address of the memory word which was the best match. Usually the address has been coded in the learning process to correspond to the letter D.

Now the system has been programmed to reexamine the three memory words which matched within ten units of the input word. Suppose that of these three words, two correspond to the same output (let's say "D"). The system retains information about this redundancy. Such redundancies are sometimes useful, but too many will be uneconomical of memory storage space. Therefore redundancies are programmed for removal either when (1) the memory is overloaded, or (2) a canonical word has considerable overlap with a canonical word belonging to another set. The examination of the memory for nonuseful redundancies takes place during the reading (matching) process.

If no suitable matching (recognition) can be accomplished for a symbol, the machine changes its state from a reading mode to a learning mode and stores the canonical form of the input image. Unless an outside operator now gives this symbol a specific "name" so that the system may print out the desired name of this image, the machine will, when reading this symbol, simply print out a number corresponding to the address of the stored canonical word.

ROTATION INVARIANCE

In this system, an image can also be examined for matching with memorized canonical words when there is relative rotation of the images. The effect of a rotation is to change the row order of the canonical matrix. That is, a rotation of an image by ten degrees clockwise is identical to placing the top row of the canonical matrix on the bottom of the matrix. In a similar way, any amount of image rotation is equivalent to a corresponding amount of canonical matrix row rotation. Thus in the recognition process, it is quite simple to look for a "match" with a certain amount of rotation added to the canonical word form.

SUMMARY

The system is composed of an electronic shutter, a photoreceptor matrix (the retina), a centering system, a pulse timer, and a digital computer with a specific program. When the machine is in the learning mode, images appearing on the retina will be centered and transformed (with size invariance) to a canonical matrix which in turn may be reduced to a canonical word stored in the computer memory.

The images need not be simple alphabetical patterns, they may be human faces, navigational guideposts, or any other sort of image. For example, one may utilize an extension of our ideas to identify human faces or navigate by terrain recognition.

After a sizable memory is obtained, the machine may be put into the read mode. It is to be noted that the machine is usually in the read mode. Each incoming image is tested for a certain degree of matching with previously stored patterns. If the approximate matching result indicates that the image is not "identified" the machine will automatically go into the learning mode and store the canonical word representing the image.

The canonical word (or words for complex images) is size invariant in the sense that a large or small image will result in the same representational code. Rotation invariance occurs through testing the representation for rotation. Each canonical matrix representation can be tested for left hand rotation by removing matrix rows from the top and putting these on the bottom. Right hand rotation reverses this matrix manipulation.

The matching procedure between the memory and the image representation allows for a certain degree of "smoothing". First of all, "similar" patterns which are distorted relative to each other will occupy different memory positions. Thus there is ability to recognize misshapen patterns through utilizing additional memory space. Secondly, the matching proceeds through an elimination of mis-matches and narrowing of possibilities. The difference between the image representation and the stored canonical forms is permitted to be large (a large number of differences) at first. Then, as the possibilities are lessened, a better match is attempted. The mathematical procedure of matching is a logical procedure which can be programmed on any electronic digital machine (although some machines are easier to employ than others). It should be noted that the conventional decision procedure of electronic computers (i.e., "is A $>$ B?") is more powerful than it first appears. For example, by adding a predetermined number x to A or B one can manipulate the logic to permit a decision as to whether A is x units larger or smaller than B. This then, is a smoothed match between A and B.

The machine is also programmed to correct for overlapping sets of figures. For example, the letters U and V have rather similar canonical representations. Consequently the maximum amount of smoothing permitted for these letters will be automatically found by machine logic with allowance for the fact that very little overlap in the recognition should be tolerated. The significance of this in the recognition procedure is that the smoothing of U and V in the reading procedure will be found to be less than for other patterns which are not as similar. Such problems are entirely solved by the machine program. The determination of smoothing values are found by looking for the maximum which separates out memory units having different output signals. In other words, the machine will continue the matching procedure as long as any of the matching memory words have different outputs, but will stop when the matching memory addresses (and they may well be multiple) correspond to the same set (same output).

Since each pattern can have considerable malformations, there may be a number of memory locations corresponding to a particular output. During the course of reading, the machine is to be programmed to recognize these redundancies and eliminate them whenever they are not required to prevent overlap with other patterns. It may be well to explain this more fully. If a large smoothing is possible for some particular pattern without confusing different sets of patterns (patterns with different outputs) then there is no need for memorizing many forms of that particular (since the forms with malformations are recognized through smoothing). Consequently, the machine program eliminates stored patterns which do not help to isolate a recognition set. This program is needed if we are to utilize a machine with a small memory capacity.

Finally, there are a number of ramifications planned. Recognition of words, people, assembly line parts, etc., requires more memory capacity than letters, but the procedure is the same. It is perhaps simplest to store the canonical matrix rather than canonical words as more information is then handled in a uniform manner. With the additional information, redundancy increases. Consequently, some of the recognition problems are lessened. For example, distinguishing between U and V is much simpler when these are contained in words than when they stand alone.

ACKNOWLEDGEMENT

It is a pleasure to thank Peter M. Kelly, Lloyd Lambert, and C. L. Wanlass of the Aeronutronic Physical Electronics and Bionics Division for their interest and helpful criticism.

REFERENCES

1. J. R. Singer, "Electronic analog of the human recognition system", Journal Optical Society of America, Vol. 51, pp. 61-70, January 1961.

2. J. R. Singer and P. M. Kelly, paper given at the First Bionics Symposium, Dayton, Ohio, September 13-16, 1960.

3. J. R. Singer, "Radar Pattern Recognition", paper presented to the Radar Symposium, A.F.C.R.C., Lexington, Mass. (Vol. III of the Proceedings), October 24-28, 1960.

4. B. E. Meserve, Fundamental Concepts of Geometry, pp. 172-173, Addison-Wesley (1957).

5. S. L. Polyak, The Retina, pp. 237-242, University of Chicago Press, 1948.

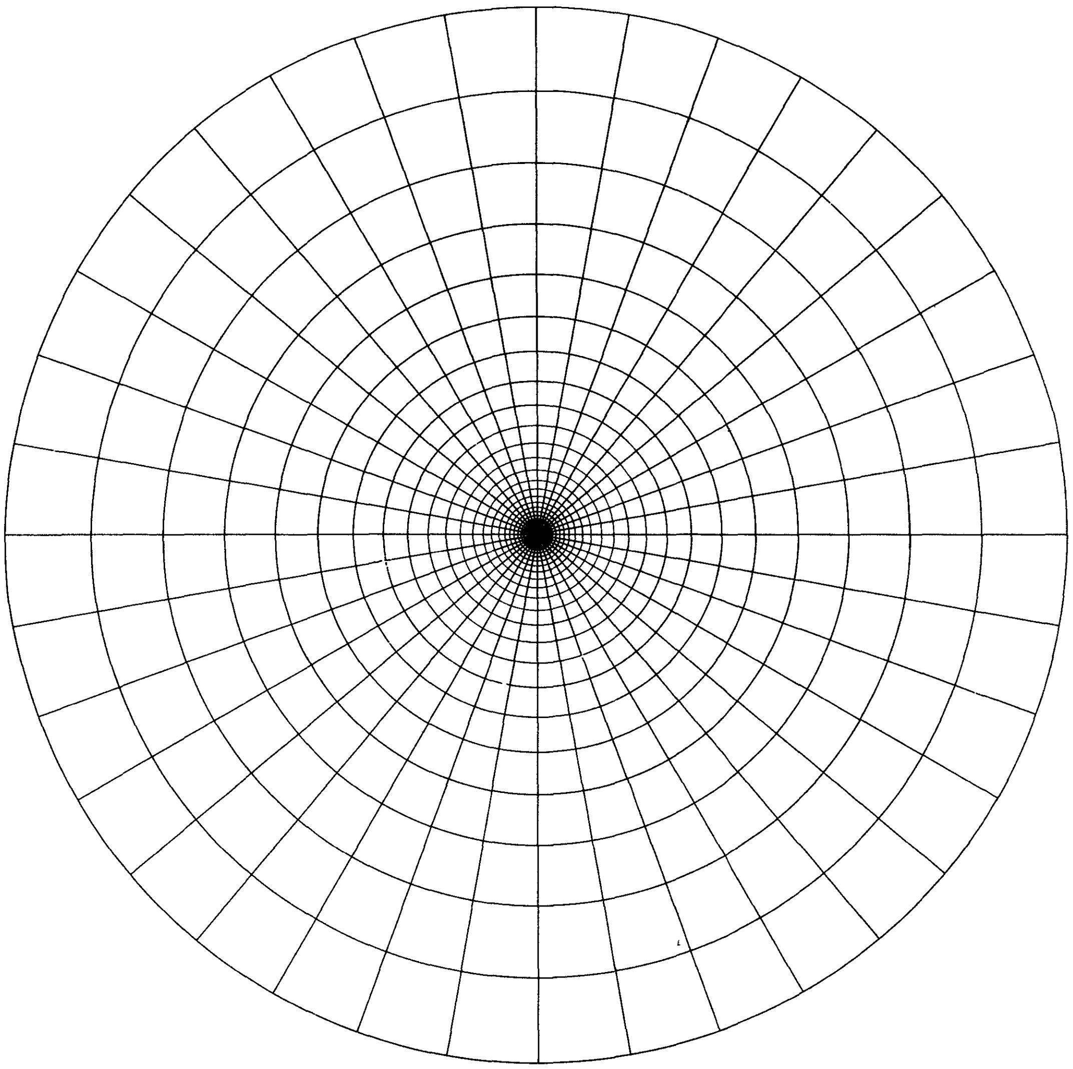


FIGURE 1

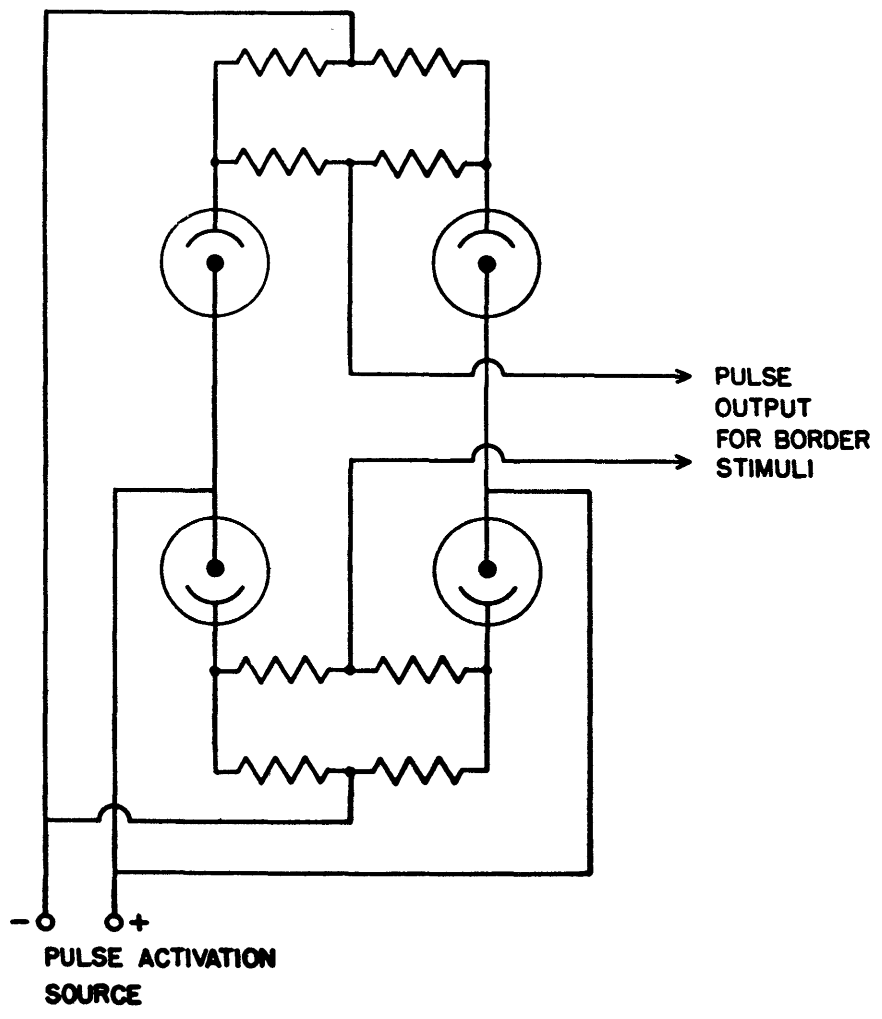


FIGURE 2

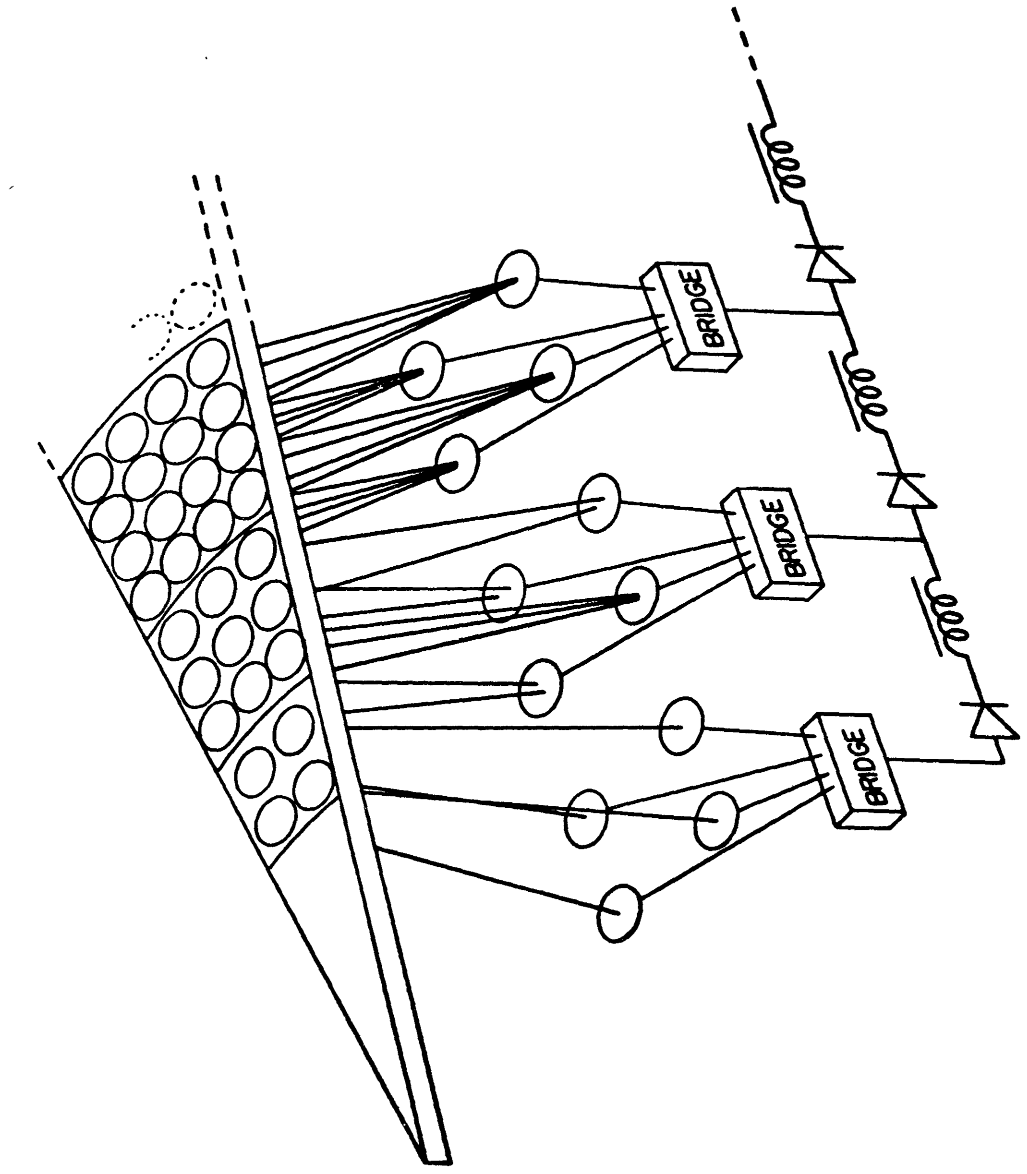


FIGURE 3

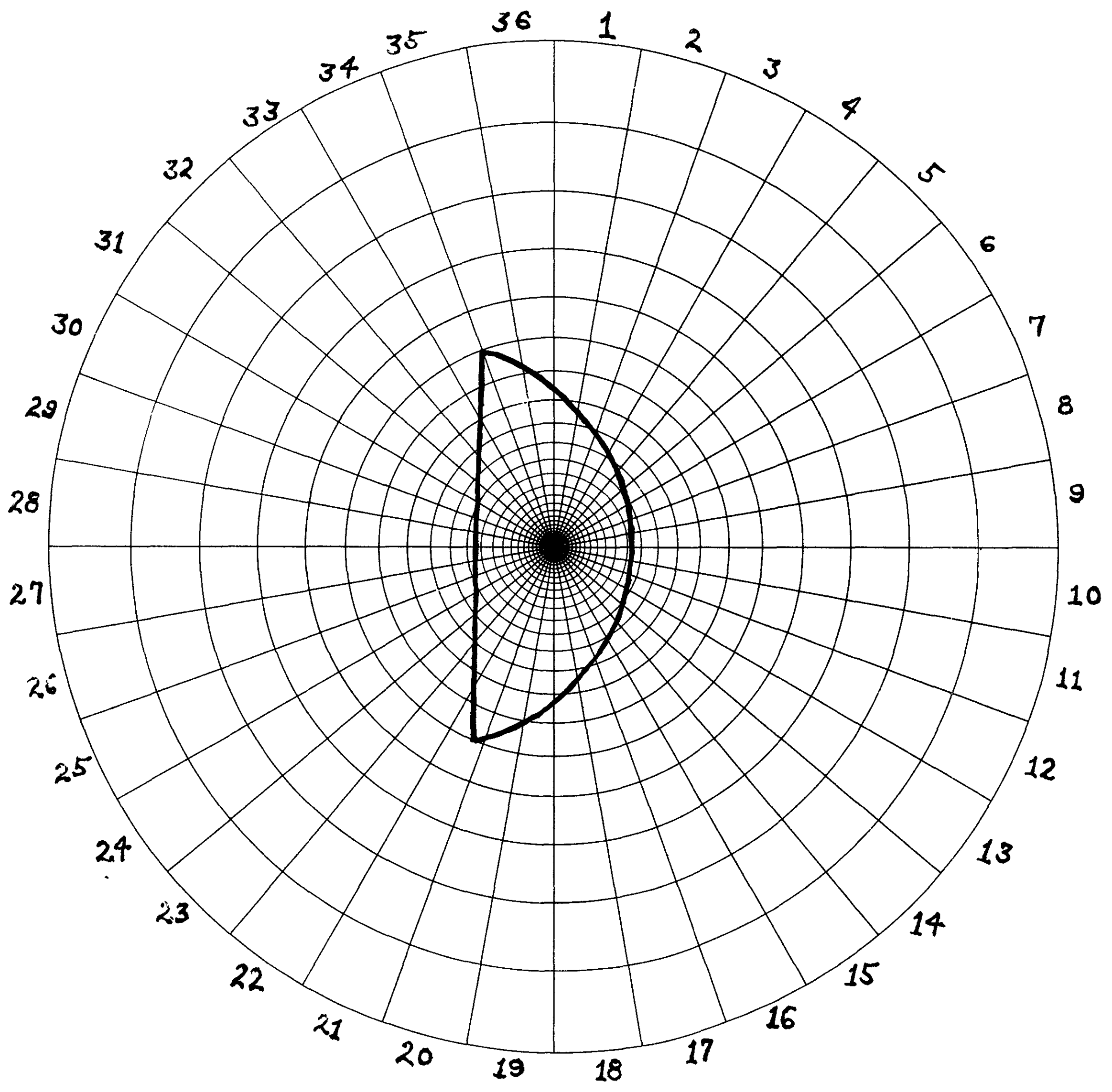

36
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35

FIGURE 4

A PATTERN RECOGNITION PROGRAM THAT GENERATES, EVALUATES, AND ADJUSTS ITS OWN OPERATORS

by

Leonard Uhr
Mental Health Research Institute
University of Michigan
Ann Arbor, Michigan

Charles Vossler
System Development Corporation
Santa Monica, California

Summary

This paper describes an attempt to make use of machine learning or self-organizing processes in the design of a pattern-recognition program. The program starts not only without any knowledge of specific patterns to be input, but also without any operators for processing inputs. Operators are generated and refined by the program itself as a function of the problem space and of its own successes and failures in dealing with the problem space. Not only does the program learn information about different patterns, it also learns or constructs, in part at least, a secondary code appropriate for the analysis of the particular set of patterns input to it.

Background Review

The typical pattern recognition program is either elaborately preprogrammed to process specific arrays of input patterns, or else it has been designed as a tabula rasa, with certain abilities to adjust its values, or "learn." The first type often cannot identify large classes of patterns that appear only trivially different to the human eye, but that would completely escape the machine's logic.[2,7] The best examples of this type are probably capable of being extended to process new classes of patterns.[8,19] But each such extension would seem to be an ad hoc complication where it should be a simplification, and to represent an additional burden of time and energy on both programmer and computer.

The latter type of self-adjusting program does not, at least as yet, appear to possess methods for accumulating experience that are sufficiently powerful to succeed in interesting cases. The random machines show relatively poor identification ability.[15,16] (One exception to this statement appears to be Roberts' modification of Rosenblatt's Perceptron.[14] But this modification appears to make the Perceptron an essentially non-random computer.) The most successful of this type of computer, to date, simply accumulates information or probabilities about discrete cells in the input matrix.[3,10] But this is an unusually weak type of learning (if it should be characterized by that vague epithet at all), and this type of program is bound to fail as soon as, and to the extent that, patterns are allowed to vary.

Several programs compromise by making use of some of the self-adapting and separate operator processing features of the latter type of program, but with powerful built-in operations of the sort used by the first type.[6, 25] They appear to have gained in flexibility in writing and modifying programs; but they have not, as yet, given (published) results that indicate that they are any more powerful than the weaker sort of program (e.g. Baran and Estrin) that uses individual cells in the matrix in ways equivalent to their use of "demons" and "operators." A final example of this mixed type of program is the randomly coupled "n-tuple" operator used by Bledsoe and Browning.[4, 5] In this program, random choice of pairs, quintuples and other tuples of cells in the input matrix is used to compose operators, in an attempt to get around the problems of pre-analyzing and pre-programming. This method appears to be guaranteed to have at least as great power as the single cell probability method.[23] But it has not as yet demonstrated this power. And it would, like most of the other programs discussed (or known to the authors) fall down when asked to process patterns which differed very greatly from those with which it had originally "gained experience" by extracting information.[22]

Summary of Program Operation

In summary, the presently running pattern recognition program works as follows: Unknown patterns are presented to the computer in discrete form, as a 20x20 matrix of zeros and ones. The program generates and composes operators by one of several random methods, and uses this set of operators to transform the unknown input matrix into a list of characteristics. Or, alternately, the programmer can specify a set of pre-generated operators in which he is interested.

These characteristics are then compared with lists of characteristics in memory, one for each type of pattern previously processed. As a result of similarity tests, the name of the list most similar to the list of characteristics just computed is chosen as the name of the input pattern. The characteristics are then examined by the program and, depending on whether they individually contributed to success or failure in identifying the input, amplifiers for each of these characteristics are then turned up or down. This adjustment of amplifiers leads eventually to discarding operators which produce poor characteristics, as indicated by low amplifier settings, and to their replacement by newly generated operators.

One mode of operation of the present program is to begin with no operators at all. In this case operators are initially generated by the program at a fixed rate until some maximum number of operators is reached. The continual replacement of poor operators by new ones then tends to produce an optimum set of operators for processing the given array of inputs.

Details of Program Operation

The program can be run in a number of ways, and we will present results for some of these. The details of the operation of the program follow.

1. An unknown pattern to be identified is digitized into a 20x20 0-1 input matrix.

2. A rectangular mask is drawn around the input (its sides defined by the leftmost, rightmost, bottommost, and topmost filled cells).

3. The input pattern is transformed into four 3-bit characteristics by each of a set of 5x5 matrix operators, each cell of which may be visualized as containing either a 0, 1, or blank. These small matrices which measure local characteristics of the pattern are translated, one at a time, across and then down that part of the matrix which lies within the mask. The operator is considered to match the input matrix whenever the 0's and 1's in the operator correspond to identical values in the pattern, and for each match the location of the center cell of the 5x5 matrix operator is temporarily recorded. This information is then summarized and scaled from 0 to 7 to form four 3-bit characteristics for the operator. These represent 1) the number of matches, 2) the average horizontal position of the matches within the rectangular mask, 3) the average vertical position of the matches, and 4) the average value of the square of the radial distance from the center of the mask.

A variable number of operators can be used in any machine run. This can mean either a number pre-set for that specific run, or a number that begins at zero and expands, under one of the rules described below, up to a maximum of 40. The string of 25 numbers which defines a 5x5 matrix operator can be generated in any of the following ways:

a. A pre-programmed string can be fed in by the experimenter.
b. A random string can be generated; this string can be restricted as to the number of "ones" it will contain, and as to whether these "ones" must be connected in the 5x5 matrix. (We have not actually tested this method as yet.)
c. A random string can be "extracted" from the present input matrix and modified by the following procedure (which in effect is imitating a certain part of the matrix). The process of inserting blanks in the extracted operator allows for minor distortions in the local characteristics which the operator matches.

(1) A 5x5 matrix is extracted from a random position in the input matrix.
(2) All "zero" cells connected to "one" cells are then replaced by blanks.
(3) Each of the remaining cells, both "zeros" and "ones," are then replaced by a blank with a probability of $\frac{1}{2}$.
(4) Tests are made to insure that the operator does not have "ones" in the same cells as any other currently used operator or any operator in a list of those recently rejected by the program. If the operator is similar to one of these in this respect a new operator is generated by starting over at step 1.

4. A second type of operator is also used. This is a combinatorial operator which specifies one of 16 possible logical or arithmetic operations and two previously calculated characteristics which are to be combined to produce a third characteristic. These operators are generated by the program by randomly choosing one of the possible operations and the two characteristics which are to be combined. This random generation process is improved by generating a set of ten operators, and then pre-testing these using the last two examples of each pattern which have been saved in memory for this purpose. This pre-testing is designed to choose an operator from the set which produces characteristics that tend to be invariant over examples of the same pattern yet vary between different patterns.

Since these operators may act upon characteristics produced by previous operators of the same type, functions of considerable complexity may be built up.

5. The two types of operators just described produce a list of characteristics by which the program attempts to recognize the unknown input pattern. At any time the program has stored in memory a similar list of characteristics for each type of pattern which the program has previously encountered. Corresponding to each list of characteristics in memory is a list of 3-bit amplifiers, which give the current weighting for each characteristic as a number from 0 to 7.

The recognition process proceeds by taking the difference between each of the characteristics for the input pattern and those in the recognition list of the first pattern. These differences are then weighted by the corresponding pattern amplifiers, and then by general amplifiers which represent the average of the pattern amplifiers across all patterns, producing a weighted average difference between the input list and the list in memory. This average difference is multiplied by a final "average difference" amplifier to obtain a "difference score" for the list in memory. When a difference score has been computed for each list in memory, the name of the list with the smallest score is printed as the name of the input pattern.

6. After each pattern is recognized the program modifies pattern amplifiers in those patterns which have difference scores less than or only slightly above the difference score for the correct pattern. This means that the program will tend to concentrate on the difficult discrimination problems, since amplifiers are adjusted only in those patterns which appear similar to the correct pattern in terms of the difference scores and therefore make identification of the input difficult. The correct pattern is compared with each of the similar patterns in turn. Each characteristic in the memory lists for a pair of patterns is examined individually, and a determination is made as to whether the correct pattern would have been chosen if the choice had been made on the basis of this characteristic alone. If this one characteristic would have identified the correct pattern, then the corresponding amplifier is turned up by one. If it would have identified the wrong pattern then the amplifier is turned down by one. If no information is given by the characteristic, for example, if it is the same for both patterns, then the amplifier is turned down with a probability of 1/8. If the pattern compared with the correct pattern had the higher difference score then the amplifiers are adjusted only in that pattern. Otherwise, amplifiers are adjusted in both patterns. This means that if several patterns obtained lower scores than the correct pattern then the amplifiers in the correct pattern will be drastically changed, since they will change when compared with each of these patterns.

The list of characteristics in memory for the pattern just processed is then modified. The first time a pattern is encountered its list of computed characteristics is simply stored in memory along with its name. On the second encounter of a pattern each of the characteristics in memory is replaced by the new characteristic with a probability of 1/2. For the third and following encounters each characteristic is replaced by the new value with a probability of 1/4. Since about 1/4 of the characteristics will be changing each time, after several examples of a pattern have been processed, the list of characteristics in memory will tend to be more similar to the characteristics of the last patterns processed than to those processed earlier. However, to the extent that the learning process is able to produce operators giving invariant characteristics for a single pattern, the list of characteristics will be representative of all the examples processed. The reason for not simply using the average value for each characteristic is that this would require saving in memory more than the 3 bits otherwise needed for each characteristic, as well as saving an indication of the number of times each characteristic had been calculated for each pattern.

An alternate scheme which we tried involved saving the highest and lowest values obtained by each characteristic, and averaging these to obtain a mean value with which to compare the input. This worked quite well in all our test runs, which used a few samples of each pattern. But there is the possibility that with large numbers of examples of a pattern, all the characteristics will eventually have very large ranges; that is, the lower bounds will tend to be 0 and the upper bounds will tend to be 7.

7. The average difference amplifiers which are used in the final step of the recognition process provide only coarse adjustments. These amplifiers are initially set to some fixed value, e.g. 60, and are then adjusted for the same pairs of patterns as the pattern amplifiers. The amplifier for the correct pattern is turned down by N if there are N incorrect patterns, and the amplifier for each of the similar patterns is turned up by one.

8. The general characteristic amplifiers are now computed by averaging the pattern amplifiers across all patterns. These indicate the general value of each characteristic in the recognition process and form the basis for the construction of success counts which control the replacement of operators. Since the combinatorial operators combine characteristics to produce other characteristics, the success count should reflect both the value of a characteristic in the recognition process and the importance of this characteristic in aiding the creation of other, possibly important characteristics.

9. This success count is formed by first storing the value of the general characteristic amplifier corresponding to each characteristic in a table for success counts. Then starting with the last combinatorial operator and working back through the list of these operators, 1/2 the value of the success count for the characteristic corresponding to the operator is added to the success counts of the two characteristics which the operator combines. Finally, two times the general characteristic amplifier setting is added to each success count.

10. Whenever a new operator is generated, the characteristics produced by the operator are computed for each of the possible patterns using the last example of each pattern, which has been saved in the computer memory. These newly calculated characteristics are then inserted into the list of characteristics for their respective patterns. At the same time the pattern amplifier settings for each of these new characteristics are set to 1 so that the characteristic will have very little weight in computing a difference score until it has been turned up as a function of proved ability at differentiation. Since the general amplifier for a characteristic is simply the average of the pattern amplifiers, it will also be 1 for the new characteristic. The success count of a new characteristic which is not combined to produce other characteristics is then 3 and this value will tend to increase if the operator proves to be valuable. On the other hand if a success count drops below 3 (or in the case of a matrix operator, if the average value of the success counts of its four characteristics drops below 3) the operator is rejected and a new operator is generated to take its place.

The pattern amplifiers play a crucial part both by aiding directly in the recognition process and by providing the information which ultimately determines the generation of new operators to replace poor ones. Since the adjustment of these amplifiers is made selectively, based on their individual success or failure in distinguishing pairs of patterns where confusion is likely, the operators rejected by the program will tend to be those which are not useful in making the more difficult discrimination. Also, because amplifiers are usually changed more drastically when the computer makes an incorrect guess, the 5x5 matrix operators will have a higher probability of being extracted from unrecognized patterns. Although the rules governing the learning process seem rather arbitrary in many cases, and it is difficult to describe their effects quantitatively, qualitative effects, such as this ability to concentrate on difficult problems, are fairly easy to show. The description of the program's operation shows that the emphasis is not so much on the design of a specific problem solving code as it is on the design of a program which, at least in part, will construct such a problem solving code as a result of experience.

It is interesting to note that the memory of the program exists in at least three different places: 1) in the lists of characteristics in memory, 2) in the settings of the various amplifiers, and 3) in the set of operators in use by the program. While the lists of characteristics bear some direct relationship to the individual patterns processed by the program, the values of the amplifiers and the set of operators in use by the program depend in a more complex way on the whole set of patterns processed by the program, and on the program's success or failure in recognizing these patterns. The learning in the first case, which involves simply storing characteristics in memory, is merely "memorization" or "learning by rote." In the second case, the learning is more subtle for it involves the program's own analysis of its ability to deal with its environment, and its attempts to improve this ability.

Test Results

The program was written for the IBM 709 and required about 2000 machine instructions. The time required to process a single character was about 25 seconds when 5 different patterns were used and 40 seconds when each character had to be compared with ten possible patterns in memory. While such times are not excessive, they are large enough to make it impractical to run extremely large test cases.

In several early runs which we made, 48 pre-programmed matrix operators were used. These were designed to measure such things as straight and curved lines, the ends of vertical and horizontal strokes, and various other features. The program was tested using seven different sets of the five hand-printed characters A, B, C, D, and E. These involved a fair amount of distortion, and variation in size, but were not rotated to any great extent.

The program's performance on the last three or four sets in a run varied from about 70% to 80% depending on various changes which were made to the rules governing the learning process. Although the effects of the various rules were not extensively tested, the program's ability to recognize characters seemed quite dependent on the manner in which pattern amplifiers were adjusted. Originally, the amplifier for a characteristic had been turned up by 1 with a probability of 1/2 if the characteristic individually identified the correct pattern, and the amplifier had been turned down by 1 if it gave no information. Performance seemed to improve when this rule was changed so that the amplifier was always turned up for a correct response of the characteristic, and turned down only with a probability of 1/8 when no information was given by the characteristic. The first of these runs showed that few new matrix operators were being generated by the program, so changes were also made in the way the success counts were formed in order to increase the number of new operators generated.

One run was made which did not use the matrix operators. Instead, the individual cells of the 20x20 matrix were used as the first 400 characteristics and 500 combinatorial operators were used, to produce a total of 900 characteristics which the program used to recognize characters. With these changes the program recognized only a little more than 30% of the characters. The amplifier settings appeared to be generally somewhat higher for the characteristics produced by the combinatorial operators than for the input cells themselves. This indicated that the program might have done slightly, though probably not substantially, better if the recognition had been based only on these latter characteristics.

The last runs were made without pre-programmed operators, but with the program generating all operators from the start. A maximum of 40 matrix operators were used at any one time, and 160 of the combinatorial operators were used in addition to these. On a run with the same seven sets of five characters used in previous tests, the program recognized 86% of the characters correctly in sets 2 through 7. (The first set can never be correctly identified by the program, of course, since the program must always predict the name of some pattern whose characteristics it already has in memory.) In this run, the same sets of characters were then processed by the computer a second time, and in this case the program recognized 94% of them, missing only two of the 35 characters.

In another run three passes were made through three sets of the first ten alphabetic characters. In this case the program recognized 29 out of the 30 characters, or 97%, on the third encounter. With this rather limited training the program was then able to recognize 70% of the hand printed characters in a fourth set different from the three sets with which it was trained. In this case, the program's ability to recognize unknown characters was considerably less than its ability to recognize previously processed characters. This can be explained by the fact that three examples of each pattern contain only a few of the possible variations of a character. It can be expected that as the number of examples with which the program is trained increases, its ability to recognize unknown characters will also increase.

We have not made any test runs of this program on the entire 26 letter alphabet, because of the computer time involved. We have, however, made some preliminary runs in debugging a modified program that was designed to increase the speed of processing by a factor of 10 or greater, along with a number of other changes. These runs gave preliminary processing abilities around 80%, using three sets of the alphabet, after only two passes. We anticipate that the completely debugged program, with improvements in such things as amplifier adjustments plus longer runs should raise this figure.

The program was also tested using line drawings representing a chair, a table, two different faces, and two types of particle decay similar to those shown in bubble chamber pictures. Two sets of these 6 drawings were used, with the second set drawn somewhat differently from the first set. After the first set was processed, 50% of the drawings in the second set were recognized correctly by the program. When the same two sets were then processed by the program a second time, all of the drawings were correctly recognized.

Discussion

When this program is given a neurophysiological interpretation, or a neural net analog, it can be seen to embody relatively weak, plausible, and "natural-looking" assumptions. The 5x5 matrix operator is equivalent to a 5x5 net of input retinal cones or photocells converging on a single output, with "ones" denoting excitatory and "zeros" denoting inhibitory connections, and the threshold for firing the output unit set at the sum of the "ones." Each translation step of the operator matrix over the larger matrix gives a sequential simulation of the parallel placement of many of these simple neural net operators throughout the matrix. Each different operator, then, is the equivalent of an additional connection pattern between input cones, firing onto a new output unit that computes the output for that operation. This is all quite plausible for the retina as known anatomically, with a single matrix of cones in parallel that feed into several layers of neurons. Evidence for excitatory and inhibitory connections is also strong.[9] And there is even beginning to be evidence of several types of simple net operators that exist in parallel iterated form throughout the retinal matrix (four of these as determined by Lettvin, Maturana, McCulloch and Pitts in the frog; and probably even more as determined by Hubel and Wiesel in the cat).[11, 12]

It would seem, however, that the known physiological constraints and the plausible geometric constraints on operators would suggest fewer than the 40-odd operators that we have used (or than the 30-odd used by Doyle or the 75 used by Bledsoe and Browning - ignoring the fact that they cannot be so easily interpreted neurophysiologically).[4, 6] For example, straight line and sharp curve operators would seem to be more plausible in terms of the ease of connection and the importance of the information to which they respond. A possible operator that might overcome this problem, with which we are now working, is a simple differencing operator that will, by means of several additional layers of operations, first delineate contour and then compute successively higher order differences, and hence straightness, slope and curvature, for the unknown pattern. This operator appears to be equivalent to a simple net of excitatory and inhibitory elements.[24]

This, then, suggests that the mapping part of the program would be effected by two layers of parallel basic units in a neuron net-like arrangement. The matching part might similarly be performed by storing the previously mapped lists in a parallel memory and sweeping the input list, now mapped into the same standard format, through these lists. Finally, the amplifiers can be interpreted as threshold values as to when the differences thus computed lead to an output. The specific pattern characteristic amplifier would be an additional single unit layer lying right behind the memory list; the interpretation of the general amplifiers might be made in terms of chemical gradients, but is more obscure.

Thus a suitable parallel computer would perform all of the operations of this program in from three to five serial steps. This is a somewhat greater depth than those programs, such as Selfridge's and Rosenblatt's, that attempt to remain true to this aspect of the visual nervous system.[15, 17] But it is well within the limits, and actually closer to the specifications, of that system. It also takes into consideration the very precise (and amazing) point-to-point and nearness relations that are seen in the visual system, both between several spots on the retina or any particular neural layer, and from retina to cortex.[20] It also is using operators that seem more plausible in terms of neural interconnections - again, in the living system, heavily biased toward nearness.

The size of the overall input matrix has also been chosen with the requirements of pattern perception in mind. Good psychophysical data show clearly that when patterns of the complexity of alphanumeric letters are presented to the human eye, recognition is just as sure and quick no matter how small the retinal cone mosaic, until the pattern subtends a mosaic of about the 20x20 size, at which time recognition begins to fall off, in both speed and accuracy, until a 10x10 mosaic is reached, at which point the pattern cannot be resolved at all. This further suggests something about the size of the basic operator, when we consider that most letters are composed of loops and strokes that are on the order of 1/2 or 1/4 of the whole. For our present purposes, the advantage of the 5x5 operator was not only its plausibility but also the fact that it cuts down to a workable size the space within which to generate random operators of the sort we are using when we permute through all possible combinations of the matrix. Again, with the constraint that these random operators be connected, it becomes a more powerful geometry - and topology-sensitive operator, and also a simulation of a more plausible neural net.

Finally, psychophysical evidence also strongly suggests that the resolving power of the human perceptual mechanism is on the order of only two or three bits worth of differentiation as to dimensions of pattern characteristics - things such as length, slope, and curvature.[1, 13, 21] This, again, suggests a 5x5 matrix as a minimum matrix that is capable of making these resolutions.

The specifications for and methods used by living systems, and especially the human visual system, suggest certain design possibilities for a pattern recognition computer; but they certainly do not suggest the only possibilities. Nor should they be slavishly imitated. They should, however, be examined seriously, for the living pattern recognizers are the only successful systems that we know of today. Nor does it seem that the sort of use we have been making of these human specifications will impose any fundamental limitation on a program such as this, one that generates and adjusts its own operators. We have, in fact, already found the program making a different, and, apparently, more powerful, choice of operators than the choice suggested to us by the psychophysiological data and conjectures we have just described. The program's "learning" methods can now depend both on built-in connections (maturation) and on the inputs that need to be learned. The program will develop differently as a function of different input sets. It appears to be capable of extracting and successfully using information from these sets. This would seem to be as completely adaptive - being adaptive to inputs - as a computer or organism can be expected to be.

One of the most encouraging things about this program is that it still has a lot to learn. Its present level of success was achieved without any great sophistication in choice of operators or any great ability on the part of the program to generate and improve operators. More important, the program itself is in a position to improve on whatever choices it, or its programmers, make for it, and to make and to evaluate its own choices. This is so because it learns, and continues to learn, as it performs. There is good reason to feel that the present results reflect only the beginning to this process, since the program is still adjusting its set of operators and their values. The program, as it continues to run and be tested, should continue to improve. The program will be doing the improving - "learning," if you will - and not the programmers. It will be something of an experimenter on its own; and it should, in fact, present us with results, as it evolves toward "best" sets of operators, that will be the equivalent of parallel experiments between, and throw light on alternate, pattern recognition methods. For most pattern recognition schemes are close to being equivalents to one or another of the sets of operators that the present program can handle, except for those programs which make use of more complex analytic methods.

This sort of design would seem to have some applicability to a variety of more "intelligent" machines. The program replaces the programmer-analyst by a programmed operator that first generates operators that make effective enough use of the unknown input space, and then makes use of feedback as to the success of these new operators in mapping unknown inputs in order to increase their effectiveness. Thus neither programmer nor program needs to know anything specific about the problem ahead of time. The program performs, as part of its natural routine, the data collection, analysis, and inference that is typically left to the programmer. This would be a foolish waste of time for a problem that had already been analyzed. But pattern recognition, and many other problems of machine intelligence, have not been sufficiently analyzed. The different pattern recognition programs are, themselves, attempts to make this analysis. As long as pattern recognition remains in the experimental stage (as it must do until it is effectively solved), a program of this sort would seem to be the most convenient and flexible format for running what is, in effect, a continuing series of experiments upon whose results continuing modifications of theories are made. This becomes an extremely interesting process for the biologist or psychologist, especially to the extent that the program can be interpreted either physiologically or functionally, or at the least does not violate any known data. For the experimentation and concomitant theory building and modification being undertaken today is rapidly building what appears to us to be the first relatively firm and meaningful theoretical structure - for pattern, or form, perception - for the science of "higher mental processes."

Self-generation of operators, by the various methods employed in this program, may also suggest approaches toward solving a wide variety of pattern recognition and pattern extraction problems. Thus there is some hope that relatively powerful operators are being extracted and generated as a result of experience with and feedback from the program's quasi-experimental analysis on the body of data that is available to it - its inputs and the consequences of its actions. Further, the level of power of these operators, and the serial ordering of operators can also be placed under similar control. Thus operators need not be overly simple or random to be machine-chosen; nor pre-programmed to be powerful. Rather, they can arise from the problem, and thus be sensitive to the problem, and to changes in the problem.

References

1. Alluisi, E. A. "Conditions Affecting the Amount of Information in Absolute Judgements," Psychol. Rev., 1957, 64, 97-103.

2. Bailey, G. E. C. and Norrie, G. O. "Automatic Reading of Typed or Printed Characters," Brit. Inst. Radio Eng. Conv. on Electronics in Automation, 1957.

3. Baran, P. and Estrin, G. "An Adaptive Character Reader," Paper presented at IRE WESCON, Los Angeles, August, 1960.

4. Bledsoe, W. W. and Browning, I. "Pattern Recognition and Reading by Machine," Proc. Eastern Joint Comp. Conf., 1959, 225-232.

5. Bledsoe, W. W. "Further Results on the N-tuple Pattern Recognition Method," IRE Trans. Electronic Computers, in press.

6. Doyle, W. "Recognition of Sloppy, Hand-printed Characters," Proc. West. Joint Comp. Conf., 1960, 133-142.

7. Greanias, B. C., Hoppell, C. J., Kloomok, M., and Osborne, J. S. "The Design of the Logic for the Recognition of Printed Characters by Simulation," IBM J. Res. Development, 1957, 1, 8-18.

8. Grimsdale, R. L., Sumner, F. H., Tunis, C. J., and Kilburn, T. "A System for the Automatic Recognition of Patterns," Proc. IEE, Part B, 1959, 106, 210-221.

9. Hartline, H. K. "The Response of Single Optic Nerve Fibers of the Vertebrate Eye to Illumination of the Retina," Amer. J. Physiol., 1938, 121, 400-415.

10. Highleyman, W. H. and Kamentsky, L. A. "Comments on a Character Recognition Method of Bledsoe and Browning," IRE Trans. Electronic Computers, 1960, Vol. EC-9, 263.

11. Hubel, D. H. and Wiesel, T. N. "Receptive Fields of Single Neurons in the Cat's Striate Cortex," J. Physiol., 1959, 148, 574-591.

12. Lettvin, J. Y., Maturana, H. R., McCulloch, W. S., and Pitts, W. H. "What the Frog's Eye Tells the Frog's Brain," Proc. IRE, 1959, 47, 1940-1951.

13. Miller, G. A. "The Magical Number Seven, Plus or Minus Two: Some Limits on our Capacity for Processing Information," Psychol. Rev., 1956, 63, 81-96.

14. Roberts, L. G. "Pattern Recognition with Adaptive Network," IRE Conv. Rec. 1960, Vol. 8, Part 2, 1, 66-70.

15. Rosenblatt, F. The Perceptron. A Theory of Statistical Separability in Cognitive Systems. Buffalo: Cornell Aeronautical Laboratory, Inc., Report No. VG-1196-G-1, 1958.

16. Rosenblatt, F. "Perceptron Simulation Experiments," Proc. IRE, 1960, 48, 301-309.

17. Selfridge, O. G. "Pandemonium: A Paradigm for Learning," In: Mechanization of Thought Processes. London: HMSO, 1959, 511-535.

18. Selfridge, O. G. and Neisser, U. "Pattern Recognition by Machines and Men," Sci. Amer., 1960, 203, 60-68.

19. Sherman, H. "A Quasi-topological Method for the Recognition of Line Patterns," In: Information Processing. Paris: UNESCO, 1960, 232-238.

20. Sperry, R. W. "Mechanisms of Neural Maturation," In: (S. S. Stevens, Ed.) Handbook of Experimental Psychology. New York: John Wiley and Sons, Inc., 1951, 236-280.

21. Uhr, L. "Machine Perception of Printed and Hand-written Forms by Means of Procedures for Assessing and Recognizing Gestalts," Paper presented at ACM Meeting, Boston, 1959.

22. Uhr, L. "'Pattern Recognition' Computers as Models for Form Perception," (draft), Ditto, 1960.

23. Uhr, L. "A Possibly Misleading Conclusion as to the Inferiority of One Method for Pattern Recognition to a Second Method to Which it is Guaranteed to be Superior," IRE Trans. Electronic Computers, 1961, in press.

24. Uhr, L. and Vossler, C. "Suggestions for Self-adapting Computer Models of Brain Functions," Behav. Sci., 1961, in press.

25. Unger, S. H. "Pattern Recognition and Detection," Proc. IRE, 1959, 47, 1737-1752.

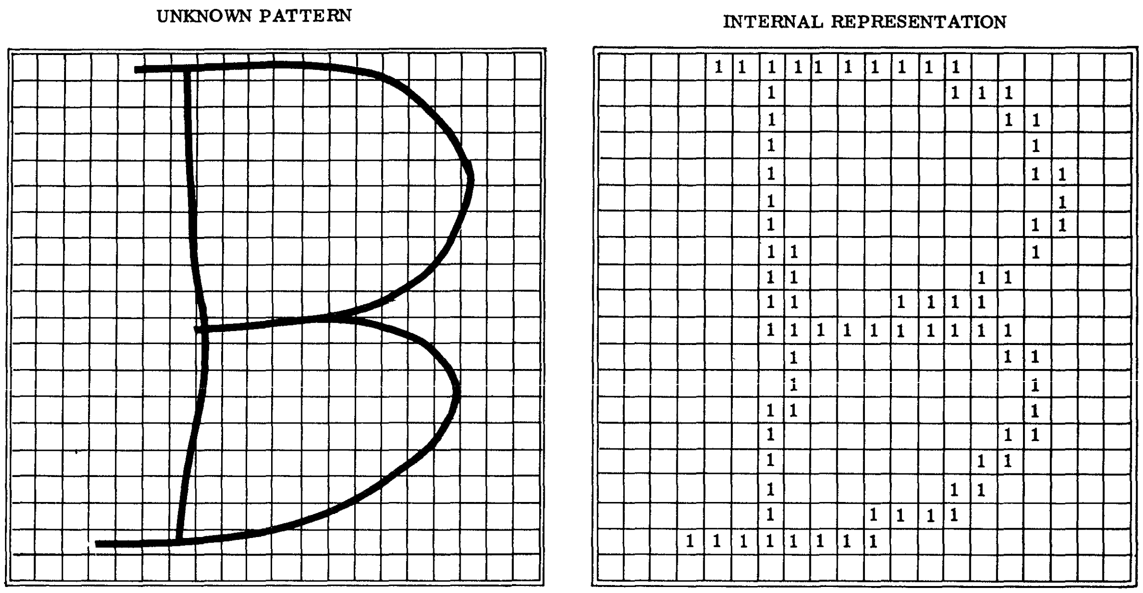


Figure 1 An unknown pattern is input as a 20 x 20 matrix with the cells covered by the pattern represented by '1's' and the other cells by '0's."

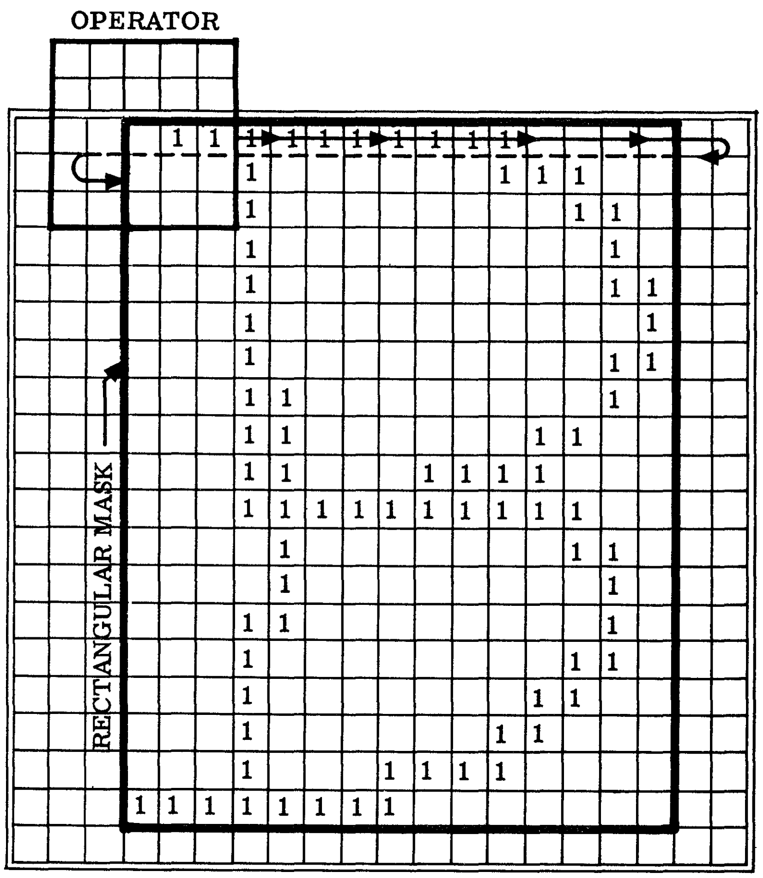
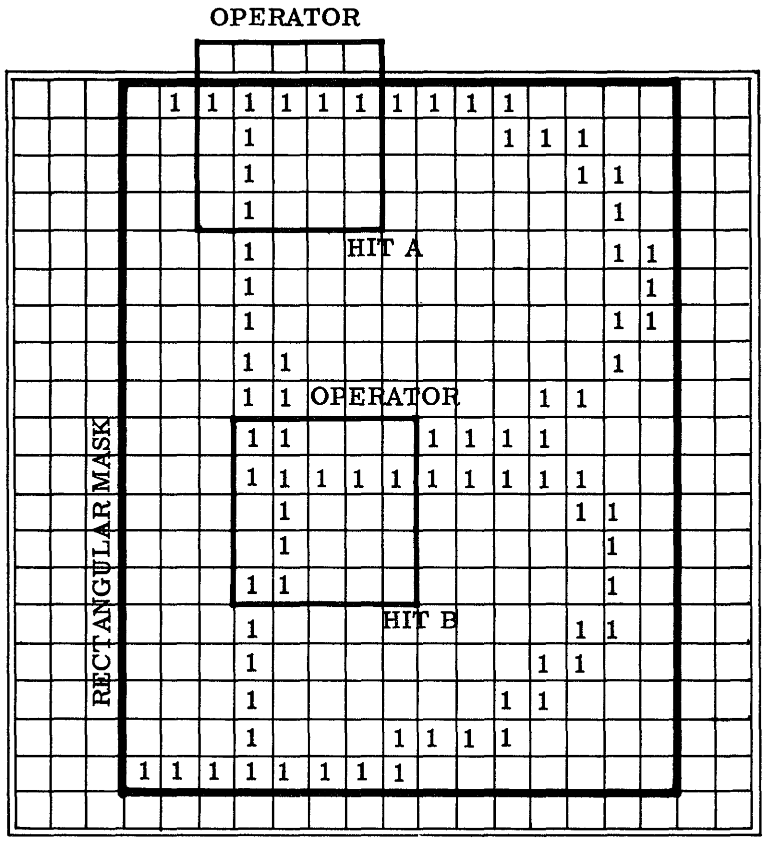
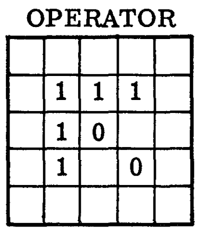


Figure 2 A rectangular mask is drawn around the unknown pattern. Each of the 5 x 5 matrix "operators" is then translated over the pattern.



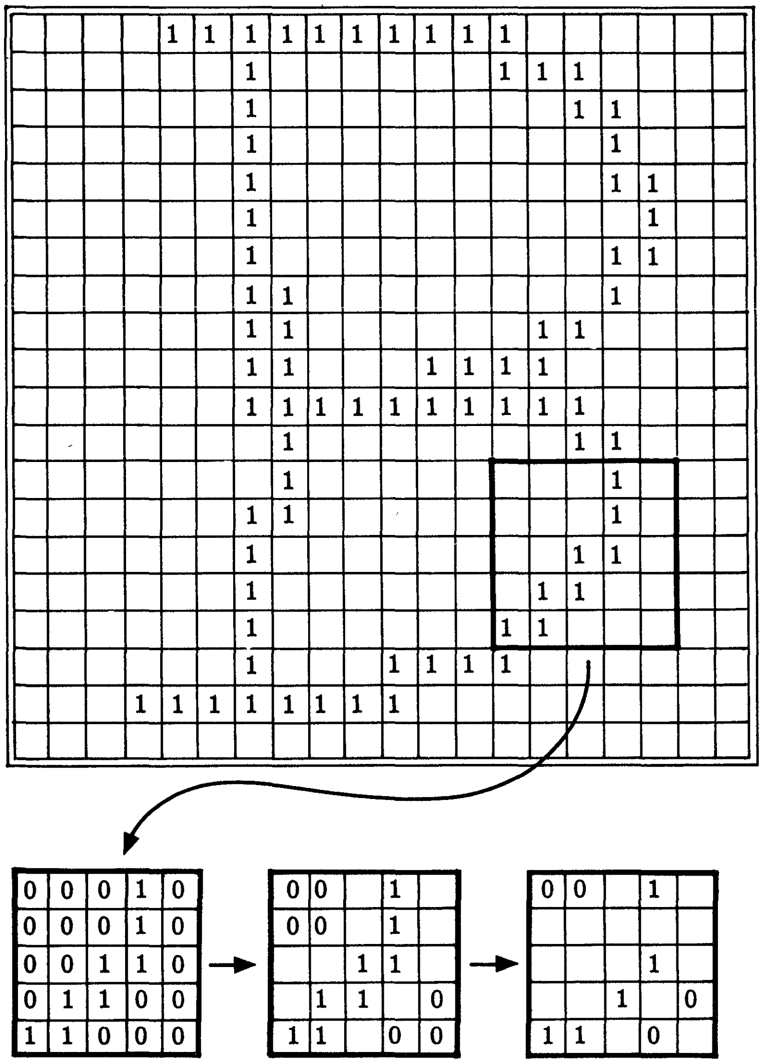


HIT	X	Y	R^2
A	1	0	4
B	2	4	0
N = 2	2	2	2

Figure 3 The operator at the lower left in the figure is shown in the two positions where it matches the input matrix. An operator gives a positive output each time its "1's" cover "1's" and its '0's" cover "0's" in the unknown pattern.

OPERATOR GENERATION

a) BY EXTRACTION



b) BY RANDOM CHOICE OF CELLS

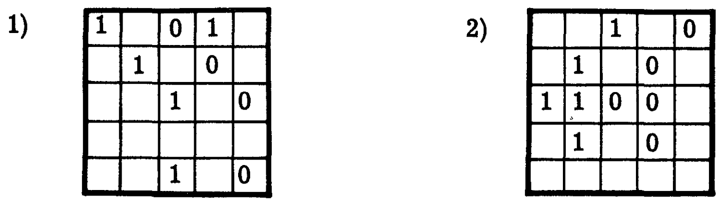


Figure 4 Operators are generated within the 5 x 5 matrix by either: a) extraction from the input pattern (random placement of a 5 x 5 matrix, elimination of "0's" connected to "1's," and elimination of each of the remaining cells with a probability of $\frac{1}{2}$) or b) by random designation of cells as either "0" or "1" (choose a "1," then place a "0" two cells to its right). In 1) from 3 to 7 "1's" are chosen completely at random, while in 2) the choice is limited to connected cells.

OPERATORS USED

A. PRE-PROGRAMMED SET

	1	1	1	
	1	0		
	1		0	

0		1		0
0		1		0
0				0
0				0
0	0	0	0	0

1				
	1			
		1		
			1	
				1

	1			
		1		
			1	
		1		
	1			

1				
1				
1	1	1		
1				
1				

				1
		1		
1				

B. SET GENERATED BY PROGRAM

			0	
	1		0	
				0
	1			0

0				
	1		0	
1			0	
1				0

	1	1	1	
			1	
0			1	
			1	
	0			

0		1		0
0				0
0				
		1		1

1	1	1	1	1
		0		
		0		0

0	0		1	
0	0		1	
0				1
		0		
	0	0		

Figure 5 A) Some typical examples of pre-programmed operators are shown. B) Six of the operators generated by the program, during a run that reached 94% success on 7 sets of 5 patterns, are shown.

	MATRIX OPERATORS								COMBINATORIAL OPERATORS			
PATTERN	OPERATOR 1				OPERATOR 2					CHARACTERISTIC		
NAME	N	X	Y	R^2	N	X	Y	R^2	. . .	m-2	m-1	m
?	2	2	2	2	2	3	1	6	. . .	6	7	4
A	3	3	4	1	4	0	1	1	. . .	1	2	3
B	2	2	3	2	3	3	2	5	. . .	4	7	5
C	4	5	6	5	1	0	0	4	. . .	2	1	7

Figure 6 Operator outputs are listed for the unknown pattern in the same format as in lists stored in memory.

PATTERN A

CHARACTERISTICS (A)	:	3	3	4	1	4 . . .	3
INPUT (?)	:	2	2	2	2	2 . . .	4
DIFFERENCE |A - ?|	:	1	1	2	1	2 . . .	1
PATTERN AMPLIFIERS	:	2	3	1	2	0 . . .	3
GENERAL AMPLIFIERS	:	3	3	1	1	0 . . .	3
DIFF. X AMPLIFIERS	:	6	9	2	2	0 . . .	9

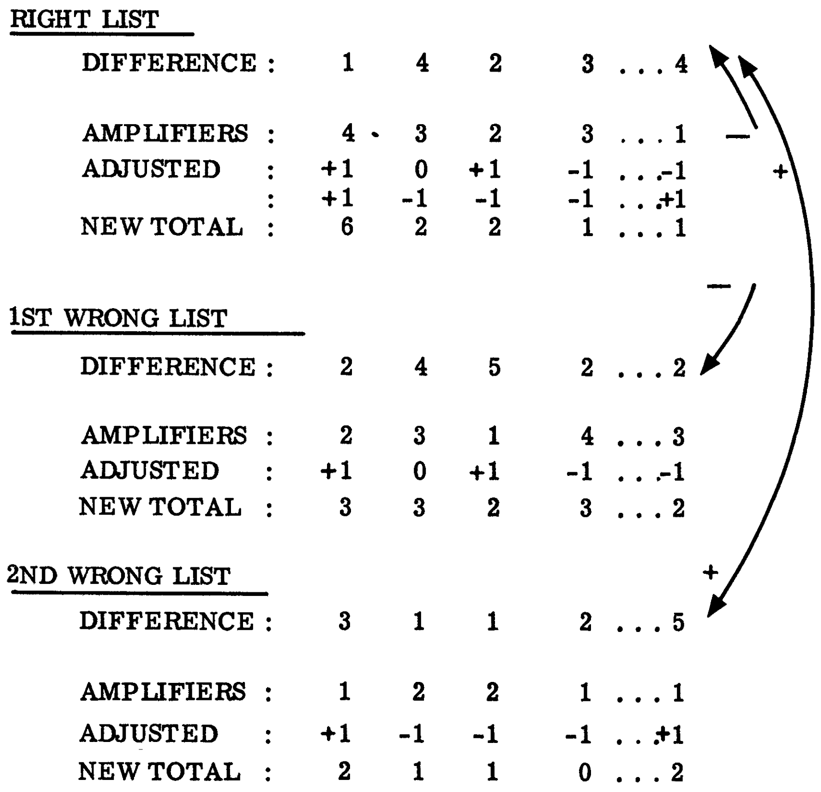
WEIGHTED AVERAGE DIFF.	AVE. DIFF. AMPLIFIER	DIFFERENCE SCORE
$\frac{28}{27}$ = 1.04	61	63

PATTERN B

CHARACTERISTICS (B)	:	2	2	3	2	3 . . .	5
INPUT (?)	:	2	2	2	2	2 . . .	4
DIFFERENCE |B - ?|	:	0	0	1	0	1 . . .	1
PATTERN AMPLIFIERS	:	4	3	2	3	2 . . .	2
GENERAL AMPLIFIERS	:	3	3	1	1	0 . . .	3
DIFF. X AMPLIFIERS	:	0	0	2	0	0 . . .	6

WEIGHTED AVERAGE DIFF.	AVE. DIFF. AMPLIFIER	DIFFERENCE SCORE
$\frac{8}{32}$ = .25	60	15

Figure 7 Differences are obtained between the characteristics for the input pattern and each list of characteristics in memory. These differences are then weighted by the product of the "general amplifiers" and "pattern amplifiers," giving a weighted average difference for each list in memory. When multiplied by corresponding "average difference amplifiers," the weighted average differences give "difference scores" for each pattern in memory. The name of the pattern with the smallest "difference score" is chosen as the name of input.



RIGHT LIST

DIFFERENCE :	1	4	2	3	. . . 4
AMPLIFIERS :	4	3	2	3	. . . 1
ADJUSTED :	+1	0	+1	-1	. . .-1
:	+1	-1	-1	-1	. . .+1
NEW TOTAL :	6	2	2	1	. . . 1

1ST WRONG LIST

DIFFERENCE :	2	4	5	2	. . . 2
AMPLIFIERS :	2	3	1	4	. . . 3
ADJUSTED :	+1	0	+1	-1	. . .-1
NEW TOTAL :	3	3	2	3	. . . 2

2ND WRONG LIST

DIFFERENCE :	3	1	1	2	. . . 5
AMPLIFIERS :	1	2	2	1	. . . 1
ADJUSTED :	+1	-1	-1	-1	. . .+1
NEW TOTAL :	2	1	1	0	. . . 2

Figure 8 The pattern amplifiers for certain lists are adjusted to increase weightings of individual characteristics that gave differences in the right direction, and to decrease weightings that gave differences in the wrong direction

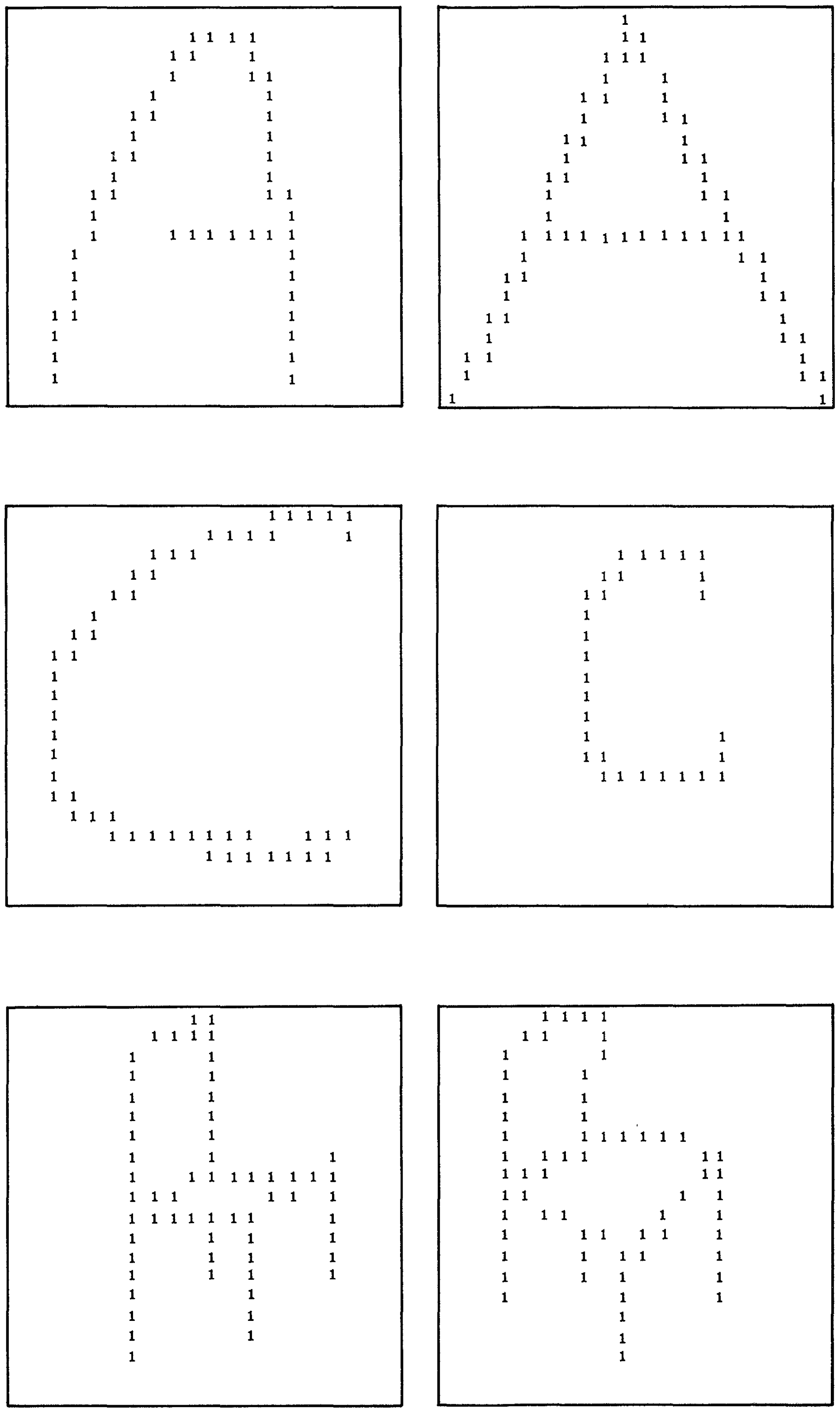


Figure 9 Two examples of an "A" and a "C" and the two line drawings of a chair are typical of unknown patterns processed.

AN EXPERIMENTAL PROGRAM FOR THE SELECTION OF "DISJUNCTIVE HYPOTHESES"

Manfred Kochen
IBM Research Center
Yorktown Heights, New York

An algorithm for finding the characterization of a class of objects on the basis of a randomly ordered sequence of labeled individual objects - some members of the class, some not - is described. The class is characterized as a disjunction of terms, each term being a conjunction of attributes. "All red, round objects or all square, small objects" is an example. Mechanisms based on this algorithm are described in terms of such properties as the amount of storage available for recording instances and the number of instances which had to be examined until the class was first guessed.

Introduction

Suppose that you were shown a red, round and small object and told, by an instructor, that this was an instance of some class of objects he had in mind; that you were required to determine this class of objects or "concept." Suppose that the instructor then selects a second object at random, which happens to be red, square and small, and he tells you that it, also, is an instance of what he has in mind. The next object he shows might be blue, round and small and not an instance of what he has in mind. The next could be red, round and large and a negative instance again. You might, perhaps, conjecture that what the instructor had in mind was the "class of all red and small objects, with shape an irrelevant attribute."*

This experimental paradigm has been used to study "hypothesis-formation"[2,4] in human thinking; a phrase such as the above in quotes marked by (*), has been called a "conjunctive concept." The experimental paradigm described above has also been the basis for simulations by computer [1,4,5] for various purposes. In a previous paper [5], to which this is a sequel, our purpose was to describe a series of mechanisms, in terms of the amount of storage they had available to record past instances and in terms of the number of instances which had to be examined before a good "guess" was obtained; we related these properties of the mechanism to the "magnitude" and "complexity" of the "concept" to be found. In this paper, this investigation is extended to a mechanism which can handle "disjunctive concepts," as exemplified by: "all red objects or all round and large objects."

The study of such mechanisms is interesting, because the number of a priori possible "concepts" grows superexponentially with the number of attributes (as 2^{2^n}), which would require an immense ** amount of storage and number of instances, if no short-cut to straightforward exhaustive procedures can be found. The mechanism to be described here is based on such a short-cut algorithm.

Formulation of the Object-Problem

For the purpose of describing the properties of our algorithm, we do not have to formulate the object-problems as a speculative model of how people would perceive them; instead, the most simply formulated object-problem would suffice. To this end, suppose that each object is completely characterized by n attributes, $x_1, \ldots, x_n$. For example, x_1 might denote color, x_2, shape, x_3, size, etc. It is further assumed that each attribute has only two values. Thus, color could only be red or blue, size large or small, etc. These two alternatives will always be denoted by 0 and 1, although it should be understood that $x_1 = 0$ does not mean the same as $x_2 = 0$. Thus, the information which is furnished to our mechanism during a single trial consists of an n-bit word and a Yes or No, indicating whether or not this object

** The term "immense" is due to Elsasser,[3] and very appropriately denotes quantities which are orders of magnitude greater than the quantities encountered in astronomy. They generally arise in combinatorial enumerations, such as is presently the case.

is judged (by the trainer) to be a member of a specified class of objects.

The classes of objects (the concept) to be identified by inference from the accumulated list of instances are denoted as follows: a set of objects for which some attribute, say x_i, is irrelevant, is denoted by the values of the attributes they have in common in the appropriate positions and an X in the ith position; thus, 01XX denotes the class of four objects, 0100, 0101, 0110, 0111. The most general "concept" is denoted by a sum of words of n characters, each 0, 1 or X. This sum represents the union of the sets denoted by the words in the sum. Thus, 01X + 1X0 + 000 denotes the class of 3-bit words which are in the set 01X or in 1X0 or 000; this is the set consisting of: 010, 011, 100, 110, 000.

It is well-known that 2^{2^n} possible subsets of the 2^n objects may be formed, although not all of them are different; for example, 01X + 00X = 0XX. Since knowledge of the possible attributes $x_1, \ldots, x_n$ is built into the mechanism, the collection of possible solutions to the problem* is well defined. In other words, our mechanisms have a priori built-in the capacity for expressing a "concept" or "hypothesis." In this sense, we are dealing with mechanisms for hypothesis-selection rather than hypothesis-formation.

Description of Algorithm

Running Example

For clarity of presentation, the following example will accompany and illustrate the algorithm description: the number of attributes is 3, and the 'concept' to be found consists of 000, 110, 111, 101, 001. This is conveniently represented on the cube of Figure 1, in which the vertices corresponding to positive instances are circled; the numbers attached to the vertices indicate the trial number or order in which the instances were received. On being informed about an instance, the mechanism delivers, as output, an hypothesis and a weight attached to this.

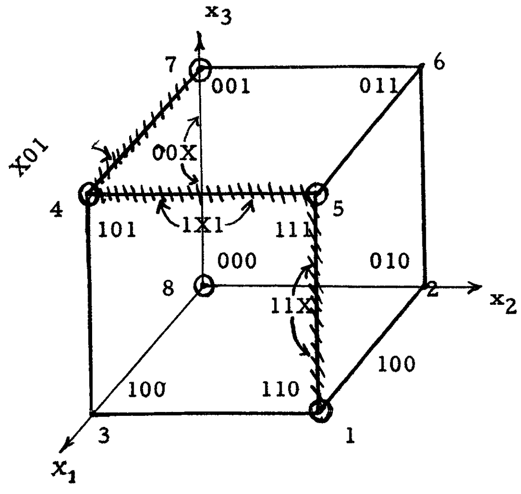


Figure 1
Running Example Illustrating the Algorithm for Sequential Selection of a Disjunctive Hypothesis

Confirming Instances

The initial hypothesis will always consist of all X's as shown in the line above trial 1 in the example in Table 1. A confirming instance, like (110, Yes) in line 1 leaves the hypothesis unchanged. As discussed in the author's earlier paper[5], there are two kinds of confirmation, and two kinds of refutation, depending on whether both the hypothesis set and the set to be found do or don't contain the current instance, and whether the instance is contained in one but not the other set.

Revising Refuted Conjunctions

A refuting instance requires a revision of the hypothesis in force. There are two types of refuting instances. These were called consistent and inconsistent (secondary) in the previous paper.[5] A refuting instance is said to be a secondary inconsistency if it leads to a logical contradiction according to the rules of inference which were valid for purely "conjunctive concepts." To illustrate this, observe that these rules of inference would lead us to deduce, from some two instances like (000, Yes) and (001, No), that the third bit must be 0; from some two instances like (000, Yes; 001, Yes), that the third attribute must be irrelevant, or X.

* By a solution to the problem is meant the specification of one of the 2^{2^n} possible sets expressible as one of the above sums, which fits some or all of the accumulated list of instances.

INPUT				OUTPUT	
Trial No.	Instance	Yes or No	Hypothesis No.	Hypothesis	Comments
				XXX	a priori.
1	110	Yes	$\cancel{①}\rightarrow\cancel{②}\rightarrow ③$	XXX	≡ hyp. no. ①.
2	010	No	$\cancel{②}\rightarrow ③ \quad \cancel{④}\begin{smallmatrix}\nearrow ⑤\\ \searrow ⑥\end{smallmatrix} \quad ⑦$	IXX	≡ ②. We deduce that $x_1 = 1$, denoted by I.
3	100	No	$③ \quad \cancel{④}\begin{smallmatrix}\nearrow ⑤\\ \searrow ⑥\end{smallmatrix} \quad ⑦$	IIX	≡ ③. Deduce further that $x_2 = 1$.
4	101	Yes	$\cancel{④}\begin{smallmatrix}\nearrow ⑤\\ \searrow ⑥\end{smallmatrix}$	IIX + XXI	XXI ≡ ④
	a positive inconsistency			add a conjunction	
5	111	Yes	$③ \quad \cancel{④}\rightarrow ⑥$	IIX + XXI	
				↙ ↘	
6	011	No	③ ⑤ ⑥ ⑦	IIX + X0I + IXI	Call X0I ≡ ⑤ IXI ≡ ⑥
	a negative inconsistency			(split into fragments)	
7	001	Yes	⑤	IIX + X0I + IXI	
8	000	Yes	⑦	IIX + XOI + IXI + OOO	Call 000 ≡ ⑦
	a positive inconsistency			add the last conjunction	

The result is ③ + ⑦ + ⑤ + ⑥, represented by the 3 shaded edges plus the origin on the cube.

Table 1
Running Example Illustrating the Algorithm for
Sequential Selection of a Disjunctive Hypothesi s

Suppose that (000, Yes) and (001, No) were observed, and now (011, Yes) is observed; the first and third instance leads to the deduction that $x_2 = x_3 = X$, while the first and second imply that $x_3 = 0$ which contradicts $x_3 = X$. Hence, the third instance is said to be inconsistent with the first two.

If an hypothesis is refuted by a consistent, negative instance and we can not logically fix a particular bit to be revised, the hypothesis in force is revised by selecting one of the X's at random and replacing it by the complement of the corresponding bit in the infirming instance. This procedure is not illustrated in the running example. In the example, the procedures followed in response to the second and third instances illustrate what is done in the case of a consistent, negative instance when the particular bit to be revised can be deduced from the current and previous instances.

If an hypothesis is refuted by a consistent, positive instance (this case does not occur in the example of Table 1), it is revised by finding that non-X bit in the current hypothesis, which, if replaced by an X, will include the infirming instance; if a single bit will not suffice, pairs of

X's, triples, etc. are tried. To illustrate, if the hypothesis in force were 0X0, and the infirming instance were (001, Yes), the third 0 would be replaced by X.

Each current hypothesis is checked for consistency with all the instances which have been recorded. To facilitate keeping track of which instances are consistent with which hypotheses, each conjunctive hypothesis is numbered according to the order in which it was first introduced. Thus XXX is assigned the number 1 (circled) in our running example. This number is placed next to every prior and future instance with which the corresponding hypothesis is consistent. (This is done only for positive instances in the computer program). If a conjunctive hypothesis becomes obsolete because it is replaced by a revision, the numbers corresponding to the original hypothesis next to each instance are crossed out. (They are actually replaced in the computer memory). The number of the revised hypothesis is placed to its right, linked by an arrow. Thus, on trial no. 3, the hypothesis 11X is in force; since it is the second revision of XXX, or the third hypothesis to be selected, it is labeled ③. Since it is consistent with all the previous three instances, its number appears uncrossed in lines 1, 2 and 3.

Adding Conjunctions

If an hypothesis is refuted by a positive and inconsistent instance, a new conjunction (term) is "disjoined" (added) to the hypothesis in force. The new conjunction is obtained by the same procedure described above, using the current (refuting) instance and all those previous instances which, taken together, will not lead to a contradiction. Obviously, not all previous instances can be used, or the current instance would not have been inconsistent. Thus, instance no. 4 (101, Yes) in our running example can belong together in a consistency class with prior instances 3 and 2 but not 1. This shows up in that the new conjunction, hypothesis no. ④, is listed in rows 4, 3 and 2, but not 1. Similarly, hypothesis no. ③ is listed in rows 1, 2 and 3, but not 4. Note also that more than one uncrossed circled number can be listed next to an instance, as in line 3.

If an hypothesis is refuted by a negative and inconsistent instance, and the hypothesis cannot be revised by modifying one of the conjunctions as described previously, then some of the conjunctions of the hypothesis in force are each split into two fragments. Each fragment contains one less X than its parent. The conjunctions to be split are all those which contain the infirming (negative) instance. In our example, the sixth instance is a case in point, and there is only one conjunction, namely XXI, which can be split. In fact, it need not be split, since it could be revised to IXI and still remain consistent, but it is split for illustration. A conjunction is split by selecting some two of its X's at random; the first fragment is formed by replacing one of the X's by the complement of the corresponding bit in the infirming instance; the other fragment is formed by replacing the other X by the complement of the bit corresponding to this X in the infirming instance. Thus, in line 6, XXI is split by (011, No) into IXI + X01. If a conjunction contains no X's, and it happens to be just the infirming instance (an unlikely event), it is simply deleted; if it contains just one X, this X is replaced by the complement of the corresponding bit of the infirming instance. When a conjunction splits into two fragments which replace it, the number of each fragment is listed next to all its confirming instances, and the number of the parent hypothesis is crossed out wherever it has occurred.

The two features of our algorithm mentioned in the above two paragraphs go beyond previous work in that they deal with the unique problems introduced by "disjunctive concepts."

At any stage, an hypothesis is satisfactory if at least one of the numbers representing the various conjunctions in the hypothesis is listed without being crossed out next to each of the instances encountered.

Weight

To compute the weight $w(t)$ to be assigned to the hypothesis selected at the t^{th} trial, denote by $c_1(t)$ the number of instances which are contained in the hypothesis set and in the set being sought and which have occurred prior to and including the t^{th} trial; and denote by $c_2(t)$ the number of instances which are excluded from the hypothesis set and from the set being sought and which have occurred prior to and including the t^{th} trial. In our running example, $c_1(7) = 4$ and $c_2(7) = 3$. Note that as defined here, $c_1(t) = t - c_2(t)$.

Further, let $r(t)$ be the fraction of positive instances encountered so far;

$r(t) = \frac{1}{t} c_1(t)$. This should not be too different, for large t, from $s(t)\, 2^{-n}$, where $s(t)$ is the cardinality (or the total number of distinct n-bit words) of the set denoted by the hypothesis in force at the t^{th} trial.

Now let

$$\hat{p}(t) = \begin{cases} r(t) & \text{if } r(t) \neq 0 \\ s(t)\cdot 2^{-n} & \text{if } r(t) = 0 \end{cases} .$$

This is a statistical estimate of the cardinality of the set being sought; the second line serves to give a reasonable and positive estimate even from a long initial run of negative instances. Beyond the rules stated in the previous section, the revision of hypotheses is further constrained as to the number of X's a revised hypothesis may contain, by the inequality,

$$| r(t) - s(t) \cdot 2^{-n} | < \frac{1}{\sqrt{t}}$$

Now we define:

$$w(t) = \frac{t}{t+1} \left[\hat{p}(t)\, c_2(t) + (1 - \hat{p}(t))\, c_1(t) \right]$$

This expresses the intuitive requirement that if our estimate of the cardinality of the set being looked for is small, a positively confirming instance should be given much weight; if the estimate is large, a negative confirming instance should be given the greater weight. The factor $t/(t+1)$ says that more weight should be given to equally confirming instances if they occur later rather than earlier. It is easy to see that if

$$c_1(t) \neq 0,$$

$$w(t) = \frac{2}{t+1} \cdot c_1(t) \cdot c_2(t) .$$

Note that, in contrast with the weighting function in the programs for purely conjunctive concepts, there is no added term corresponding to the number of bits which have been fixed by logical deduction*, because there are, in general, many ways to express a satisfactory disjunctive hypothesis. Thus, in our running example, the final hypothesis 1X1 + 00X + 11X would also have fit the data; if it is important that the number of terms be small, this would even be a better solution. Had we followed the algorithm exactly as described (without splitting XXI from line 5 to line 6), we would have obtained this result.

Discussion

Simulation Program**

To evaluate the performance of this algorithm, it is being tested, in the form of a 709 program, on a sample of "problems." By a "problem," as presented to the 709, we mean a randomly ordered sequence of instances of some fixed "disjunctive concept," like 01X + X01 for example. A number of sequences, differing only in order, corresponding to the same "concept" are given, and the following quantities are measured from the output:

(1) the average number of trials to the first time a "good" hypothesis is guessed. An hypothesis is good if it is consistent with all the recorded instances; or, if the sum of terms representing the hypothesis is either identical with the sum of terms representing the concept, or equivalent to it (e.g., 11X + X01 + 1X1 + 000 is equivalent to 1X1 + 00X + 11X). Call this sample average $\overline{N}_0(B)$, where B represents the "concept" being searched for.

(2) The average number of trials to the first jump in $w(t)$ which is "unusually large." This sample mean is denoted by $\overline{N}_1(B)$. We found, from our empirical results on simulation with conjunctive concepts,[5] that the first large jump in weight was correlated with $\overline{N}_0(B)$, and could be used as an indicator that a good hypothesis was on hand. In those experiments, "unusually large" was defined as:

* The notation for indicating that a bit in a single conjunctive term of an hypothesis has been fixed is the same as in the author's previous paper.[5] That is, when a bit is deduced to be 0, 1 or X, it is represented as O, I and Y respectively. This is useful in detecting contradictory instances to establish inconsistency.

** This 709 program is still under development. It was expected to have the empirical results on the performance of this algorithm available in time for inclusion in this paper, but various factors beyond the author's control caused delay in the completion of the simulation. It is planned to present these results orally at the WJCC meeting and perhaps to publish them separately.

$$w(t+1) - w(t) \geq \left\{ \begin{array}{l} 3 \text{ if } t = 2 \\ 2 \text{ if } t = 3 \\ 1 \text{ if } t \geq 4 \end{array} \right\} .$$

(3) the extent to which B and the hypothesis z in force at t do not coincide. This can be measured by the sample mean $\bar{d}_B(t)$, defined as the ratio of: (a) the number of instances (up to the tth inclusive) which either belong to B but not to z or to z but not to B and (b) the number of instances in either B or z. When $\bar{d}_B(t) = 1$, then z and B are disjoint, and z is as poor an hypothesis as can be. If $\bar{d}_B(t) = 0$, then $z = B$.

It is conjectured that, as in our previous work, $\bar{N}_1(B)$ and $\bar{N}_o(B)$ will again be correlated, and that $\bar{d}_B(t)$ will decrease very rapidly to zero (possibly as a negative exponential) with t.

The quantity $\bar{N}_o(B)$ is then computed for a variety of B's in order to determine how $\bar{N}_o(B)$ varies with:

n, the number of variables describing B
m, the minimal number of conjunctive terms in the sum describing B

and

$\frac{1}{m}\sum_{i=1}^{m} k_i$, where k_i is the number of X's in the ith disjunct.

Concepts with small m and many X's are expected to be simple, and $\bar{N}_o(B)$ should be small in such cases. For B's which are about "equally simple," $\bar{N}_o(B)$ may be expected to grow exponentially with n.

Next, the above relationships are recomputed with the following important modification in the algorithm. At any trial t, the mechanism can inspect only the last T instances, and as the $(t+1)^{st}$ instance is recorded, the $(t-T+1)^{th}$ instance is lost. All features of the algorithm which involved checking previous instances for consistency, confirmation, etc. now refer only to this constantly changing record; all other features are unchanged. The relationship between $\bar{N}_o(B)$ and T is estimated for a variety of B's. The nature of this relationship cannot now be anticipated. It might be worth recalling that in the case of purely conjunctive concepts, $\bar{N}_o$ did not increase significantly for T larger than a certain constant, and that this constant did not depend strongly on n.

Theoretical Considerations

In principle, the above-mentioned relations could be derived mathematically, but the simplest examples will convince anyone trying this that the number of terms that have to be enumerated grows at such a rate as to make it practically impossible for a human analyst.* For this reason, we have simulated this algorithm in a Monte-Carlo fashion. There is much to be desired in this, as in simulations generally, about the development of a firmer statistical basis for the various estimation and sampling procedures. For example, $\bar{N}_o$ is a sample mean to estimate the true mean of a random variable N_o. Can anything be said about the distribution of the random variable $\bar{N}_o$? What should the size and composition of a sample of sequences of instances be so as to permit valid inferences about the desired relations?

There are, of course, theoretical lower bounds on $\bar{N}_o$ for an ensemble of B's, depending on what is a priori known about the various B's. For example, suppose that N disjunctive concepts are considered eligible and that all are a priori equiprobable. The average number of instances which have to be examined before the concept is found must be at least $\log_2 N$. This minimum will not be obtained with a randomly ordered sequence of instances, but one which must be carefully selected. For randomly ordered, given sequences, an expression for the minimum $\bar{N}_o$ is very difficult to derive. Even then, the design of a strategy for revising hypotheses to achieve this optimal performance is as difficult a problem as finding constructive proofs to coding theorems.

Related Problems and Possible Applications

The object-problem of our mechanism is

*Even if he had the patience, time and enough writing paper to go through such a derivation, the chances of his getting the same answer twice are small.

related to the problem of determining a Boolean expression from a partial truth table. Our method differs from such techniques in that with each new line of the truth table we obtain a better approximation to a satisfactory Boolean expression. It should be of interest to compare the performance of our algorithm with some of the existing programs which are used in the design of certain switching circuits. In most applications of this algorithm, we will not be presented with a randomly ordered sequence of instances, but will have control over the order. It is therefore, an important open problem to determine the most informative (least redundant) sequence of instances. For purely conjunctive concepts, an optimal input sequence has been developed.

The algorithm described here can also be interpreted as a dynamic classification procedure using "property matrices." The element a_{ij} of such a matrix is 1 if the ith instance (e.g. cancer-patient as distinct from a non-cancer patient, a two-dimensional dot pattern from an aerial photograph labeled "airfield" as distinct from all differently labeled patterns, etc.) possesses property j (e.g. high fever, containing some rectangle consisting of 90% or more dots, etc.), and 0 if the ith instance does not have the jth property. The "concept" is then a characterization of the label (e.g. cancer, airfield) which is assigned to some of the instances by a human judge. Note that in our scheme, both positive <u>and</u> negative instances provide information, while most of the available procedures use only positive instances.

The algorithm also embodies some elements of the abstraction process, in that it could, for example, inspect the sequence of binary-coded integers (4 Yes, 7 No, 14 Yes, 19 No, 3 No, 5 No, 20 Yes, . . .) and determine that what the positive instances had in common was divisibility by two.

If our formulation had to be revised to serve as an approximate model for human hypothesis-formation, we would, first of all, have to drop the assumption that the number and kinds of attributes can be <u>a priori</u> specified. Second, each attribute would be assigned a different weight, corresponding to its importance to the individual as a characterizing feature of objects. Third, the attributes could not be treated as strictly independent, although it may be possible to determine more or less independent attributes by some such technique as factor analysis. Fourth, each attribute has its own range of values, certainly not binary in extent; each attribute may well be regarded as a concept by itself, as may each of its values (e.g. color, red). We should probably think of a hierarchy of concepts, in that an attribute or property, (e.g. "being an animal") has values (e.g. "being a dog," "being a cat," etc.) which are also attributes that have values (e.g. "being a poodle," etc.), and so on.

General Conclusions

The above paragraph indicates how limited the simple paradigm we have described in this paper seems to be as a model of real perception-conception mechanisms. Therefore, the terms "hypothesis-formation" and "concept-formation" should be used very guardedly in view of the connections with human thinking and genuine cognitive mechanisms that such terms suggest.

Nevertheless, the procedure described in this paper may serve to show another way of using computers as experimental tools in the analysis of very simple and preliminary models for cognitive mechanisms. It is also possible that the algorithm described here can be useful in certain tasks involving dynamic classifications. Furthermore, such results as the relation between $\overline{N}_0$ and T could be of sufficient generality to provide some real insight into some of the properties of memory and possibly some guidance for machine design.

References

1. Banerji, R.B. "An Information-Processing Program for Object Recognition," <u>General Systems</u>, Vol. V., Society for General Systems Research, Ann Arbor, 1960, p. 117.

2. Bruner, J.S., Goodnow, JJ., Austin, H.H. <u>A Study of Thinking</u>, Wiley, New York, 1956.

3. Elsasser, W.M. <u>The Physical Foundation of Biology</u>, Pergamon Press, New York, 1958, pp. 152-153. (See also Review of this book by M. Kochen, <u>Information and Control</u>, April, 1961.)

4. Hovland, C.I. and Hunt, E.B. "Computer Simulation of Concept Attainment," paper presented at AAAS meeting in New York, January, 1960.

5. Kochen, M. "Experimental Study of 'Hypothesis-Formation' by Computer," 1960 London Symposium on Information Theory. Butterworths, in press. Also, IBM Research Report RC-294, May, 1960.

TIME-ANALYSIS OF LOGICAL PROCESSES IN MAN

Ulric Neisser
Brandeis University
Waltham, Massachusetts

Summary

Pattern recognition in man is assumed to be mediated by a hierarchical organization of specialized systems, each of which abstracts a certain property from its input. Some characteristics of the overall organization can be inferred from the times that are needed to carry out various information processes. These times can be determined independently of reaction factors by a scanning method. Preliminary results with the method demonstrate the flexibility of the processing hierarchy in man. They also suggest that parallel processing is used under some conditions, while sequential procedures are dominant in others.

It is a commonplace now that perception and judgement involve the processing of information. The world presents itself to our sense-organs in a superficially chaotic flow of data, from which perceived objects as well as concepts and ideas are abstractions. Human behavior can be regarded as consisting of a series of decisions each of which is based on very extensive stimulus analysis carried out over time. From this point of view, the psychologist who studies cognition is examining the characteristics of a processing system.

It seems clear that the adult human perceptual and cognitive apparatus must be hierarchically organized. A system so flexibly able to respond to many properties of the input can not be radically redesigned for each task. For the most part, new stimulus analyses must occur by reorganization of existing parts rather than by starting from scratch. When you learn to recognize a new word, for example, you certainly use pre-existing and established sub-systems that identify the letters of the English language. You simply use a novel combination of their outputs. The letter-systems, in turn, are probably fed by the output of simpler organizations (let us call them "recognizers") which select various kinds of curves and shapes from the visual input.

There are three fundamental methods for exploring the organization of the processing hierarchy in man. We can look inside with the techniques of physiology; we can take another kind of look by introspecting on the processes as they occur in ourselves; or we can make inferences from human behavior. Each method has both advantages and drawbacks, and this is not the place to discuss them. The procedure reported here is based on inference from behavior, as is most of modern psychology. In particular, it takes advantage of certain time-relations in human cognitive activity. However fast the information processes may be, they must occur in real time. In situations which permit reasonable inferences about the patterns of processing, we may be able to find confirmation by looking at the times involved.

A particularly interesting question stems from the distinction, in programming, between <u>parallel</u> and <u>sequential</u> processing. As Selfridge[1] has pointed out (see also Selfridge and Neisser [2]), a device for pattern recognition may use either of two fundamental modes, which have been called <u>sequential</u> and <u>parallel</u>. They may be used singly or in combination. In the sequential mode, each partial analysis of the input results in a decision which governs the type of analysis to be made next. Only one process is carried out at a time, and the particular sequence of processes determines the final result. In the parallel mode, many analyses are made simultaneously, with the outcome depending on some (perhaps linear) combination of their outputs. How do human beings operate? It is evident from introspection and gross observation that the sequential mode is common. We often think, and act, step-by-step. On the other hand, the anatomy of the nervous system rather suggests parallel operation. It is likely that people are capable of working in either mode, depending on circumstances. The sort of time-analysis to be described here is particularly well adapted for discovering these circumstances.

Method

Fundamentally, our experiments involve timed visual pattern recognition. That is, the subject must decide as quickly as possible whether the stimulus input has a certain property or not. Typically, the input might be a letter, and the question might be whether it is the letter "Z". We assume that at least two levels of processing are involved in such a task. Certain visual characteristics are abstracted from the input -- roundness, angularity, the slopes of lines, and so on -- by sub-systems we may call shape-recognizers. The recognizer for a letter, say "Z", is some weighted combination of their outputs. For example, a certain visual pattern might produce a positive response in a recognizer for horizontal lines, and in another which detected slanted lines. These two in turn would activate the recognition system for "Z", but not that for "O". (Note that the effectiveness of these particular shape-recognizers depends on context. They would not serve to distinguish between "Z" and "A", although it is easy to imagine others that would.) To be sure, we do not assume that these particular shape-recognizers exist in any person. They are meant only as illustrative examples.

Suppose now that several letters are presented at once, and the subject is asked whether any of them is "Z". If he can process them simultaneously, in parallel, his speed will be independent of the number of letters. If he must examine them one by one, his time will increase linearly with the number of letters. To be sure, there is no doubt that the processing must be sequential if the number of letters is large. We cannot examine an entire page in a flash, if only because of the limited area on which the eye can focus. But is there a more intrinsic limitation?

Another interesting case arises if the subject is not looking for "Z" alone, but for either of two letters; say "Z" or "Q". It is evident that these two letters require different shape-recognizers. That is, two different analyses of the same visual input must be made. We could easily build a computer to make them simultaneously. But can human information-processing go on in parallel under these conditions? If so, is there a limitation on the amount of parallel activity that can be carried on? A program of research is under way to answer these questions, and some preliminary results can be presented here.

Unfortunately, the actual experiments can not be as simple as the prototype described above. The time which a human subject needs in order to indicate whether a given letter is "Z" includes much more than stimulus-analysis. The total measurable reaction time includes the response itself as well. Nor can we safely consider the total time as the sum of two parts. There are many components: the subject must fixate, must begin actual search, must decide to react, must actually respond (by speaking, or pressing a button) etc. In addition, the interrelation between these times may change if the problem is altered. For example, the final decision to press the button may be longer delayed in problems where the subject feels relatively less confident. Analysis of simple reaction times is unlikely to give adequate answers to our questions. Indeed, reaction-time analysis was common in nineteenth-century psychology, and was ultimately abandoned for these reasons.

It seems possible, however, to obtain a measure of processing time that is relatively independent of reaction factors. We have tried to achieve this by using a scanning technique instead of measuring reaction times directly. The subject is not presented with a single string of letters, but with a list of fifty strings, arranged in a column. In the entire list, only a single string has the critical property. A typical list is shown in Figure 1. In this case, the subject is to look for a "Z". As soon as the list is shown, he begins to scan down from the top. When he comes upon the item with a "Z" (in this case the 15th one down), he turns a switch. The switch stops a clock, which had been started at the instant the list was presented. Thus, the time needed to find the critical item is recorded. Of course, the number at the bottom of the list, which identifies the position of the Z, is concealed from the subject's view. To insure that he has actually found the critical item, he is instructed to turn the switch to the right if the item has a dot beside it, and to the left otherwise. Both directions stop the clock, but the experimenter can check whether the decision was correct.

When this method is used, the scanning time necessarily depends on the position of the critical item in the list. The time will necessarily be greater for lists with the "Z" nearer the bottom. If the subject is given a number of lists, each with the critical item in a different position, it becomes possible to plot the time as a function of the list-position of the item. Such a plot is shown in Figure 2. Each point in the figure represents the search time on a single list. All of them were produced by one subject in a ten-minute session, working on one problem. He simply scanned down each list until he came to a "Z". Actually, our subjects always scan 20 lists in each problem, but the first six are considered practice. From the subject's point of view, the different possible positions of the critical item occur at random, so he cannot predict in advance where his scan will end. Actually, the six practice lists always have critical items in positions 5, 6, 25, 30, 45, and 46, and the 14 lists to be used in determining T/I have their critical items in positions 9, 11, 14, 16, 19, . . . , 39 and 41. The order of presentation is randomized separately for the practice lists and the experimental lists. Thus, while the subject is kept alert to the possibility of finding the item near the very top or bottom of the list, these extreme positions are not used in the analysis of the data. There is good reason to suppose that departures from linearity will occur at the extremes, especially at the beginning.

The straight line in Figure 2 has been visually fitted to the points. It is a representative example of the extent to which our data approach linearity. The fit is generally good enough so we feel justified in treating the average time per item scanned (T/I) as a meaningful quantity, which can be directly determined from the slope of the line. (We do not imply that each item is separately fixated or processed. Even if the subject treats them in groups, T/I remains a valid measure for the comparison of scanning time across different types of lists.)

The types of logical processing that can be explored with this method include any abstractions whatever that can serve to distinguish one item in such a list from all the others. The presence of a particular letter is merely an example. The experimenter may require the presence of either of two letters, or of both, or of a particular sequence. The critical feature can also be the absence of a letter or of some logical combination of letters. In every case, the T/I being measured is for the opposite of the function being sought by the subject. If he is looking for a "Z", then all the items he scans contain no "Z", and T/I reflects the time necessary to make certain that "Z" is absent. If he is looking for the absence of "Z", each of the items scanned necessarily contains one, and T/I measures the time necessary to process it.

Procedure and Results

The present paper is a report of an exploratory experiment with this method of time-analysis. Three subjects were systematically given a variety of functions, using items of two different lengths. Although the sample is small, the results seem consistent enough, and informative enough, to justify a preliminary report. Further work is in progress to check on the findings of this study.

The design of the experiment included seven different functions, or problems. Three were positive, in the sense that the subject was looking for the first occurrence of something. These functions were "Z", "Q", and "ZvQ". In the last of these, the critical item was defined by the first appearance of either "Z" or "Q" or both together. (The "or" lists were so constructed that each type of critical item actually occurred in approximately one-third of the twenty lists.) In addition to these positive functions, we studied four others that were negative or mixed. These were "-Z", "-Q", "-ZvQ", and "Zv-Q". In the first two the subject scanned down items containing the letter in question until he found one without it. In the last two, the critical item might be distinguished either by the absence of some letter (that all the others had), or by the presence of another (that no other item had), or by both.

These seven functions were realized in two sets of lists. In one set, each item was six letters long; in the other, each had only two letters. The seven functions and two lengths yielded 14 experimental conditions. The subjects went through two conditions (i.e., scanned forty lists) at a session, which took about half an hour. The first two sessions were devoted to practice with varying types of lists. Thereafter, each subject worked in each condition twice. To control for the effects of order and practice, the 14 conditions were first given in a certain order (different for each subject) and then repeated in reverse order. The results have been plotted, and slopes calculated for the best-fit lines.

The lists were prepared by an IBM 7090 computer, and printed on an ANELEX. In preparing the lists, the computer program formed random permutations of the letters J, P, Q, S, T, V, X, Z, and examined the last six (or two) places of the permutation to see if it was an example of the desired function (e.g., if it contained "Z"). The program prepared randomly ordered lists of non-examples, inserted proper examples into the chosen list-positions, and prepared an appropriately spaced printout. The printout was cut into separate lists and pasted on to 2 x 11 cards for experimental use. An apparatus was constructed in which such a card fit under a spring-loaded door. When the door was opened by the experimenter, a timer was initiated which continued to run until the turn of the subject's switch indicated that he had found the critical item. Occasionally he overlooked it, and scanned to the bottom of the list. These trials were discarded, and the same list was re-used later in the same session.*

*The author gratefully acknowledges the assistance of Paul Weene, who designed and built the apparatus, and of Emily Carota and Arthur Warmoth, who assisted in the experiment.

The results of the experiment are displayed in Table 1. The internal consistency of the data is good, for the most part. The two replications of each condition usually yield very similar values of T/I, although two of the subjects (LS and MA) show distinct improvement with practice, with the second run generally faster than the first. The important trends can be summarized as follows:

1) "-Q" is slower than "Q"; "-Z" is slower than "Z".
2) In the positive functions, "Z" is slower than "Q".
3) 2-letter items can be scanned more quickly than 6-letter items.
4) "QvZ" takes no longer than does "Z" alone.
5) With two letters, the negative and mixed functions are all equally fast.
6) With six letters, the negative and mixed functions vary in difficulty.

Conclusions

1) In scanning for "-Z", the subject must make sure that each item he passes does contain a "Z". The recognition sub-system, or recognizer, for "Z" must be fully activated each time, and the high values of T/I reflect this requirement. In the positive function, by contrast, the Z-recognizer is not fully activated until the critical item is reached. However, the subject must scan slowly enough so that this recognizer could react with the proper input. In other words, the shape-recognizers which distinguish between "Z" and other letters must have time to act. The observed time-difference between positive and negative functions may be assumed to correspond to the different levels in the processing hierarchy which they require. The positive functions need only enough time for such recognizers as (for example) "paired horizontal lines" or "angle near top". The negative functions must have time enough for, say, "Z" itself to receive its input from such features and then to react. Perhaps we are justified in saying that any artificial pattern recognizing system, if organized in parallel, would display these time differences.

2) The two-level hierarchy becomes more articulate when we consider the difference between "Z" and "Q". Why should some letters be harder to find than others? Evidently, identification of "Z" involves different features of the visual stimulus than identification of "Q", and processing the latter is easier than the former. Without direct knowledge of the critical properties, we can only speculate about the reason. Perhaps the shape-recognizers involved with "Z" are themselves hierarchically deeper than those for "Q". Perhaps it is only that more attributes of shape must be examined for "Z", but we shall see later (No. 4, below) that this might not require an increase in time.

It would be hazardous to suppose that the letter "Q" is intrinsically easier to see than "Z". The context of alternative letters must play an important part. The shape-recognizers that suffice to distinguish "Q" from J, P, S, T, V, X, and Z might not be adequate in a context that included C, D, G, and O. The processing system must be expected to change with the context, and T/I will also change. We are now conducting an experiment to explore this point.

3) At first thought, it seems entirely reasonable that six letters should take longer than two. After all, each item contains substantially more information in the 6-letter case. Yet it would be easy to build a 6-channel device that would handle both cases with equal speed. It follows that the subjects are not acting like such a device; they do not process all the letters in parallel. There is no immediately clear reason why they should not do so. The entire six letters subtend a visual angle of less than 5°, and can easily be read in a single fixation. Thus, the visual information can all get to the cortex simultaneously. (Experiments now in progress confirm that the actual number of letters, and not their spatial separation, is the important variable.) We conclude that, at least under some circumstances, the shape recognizers can not be applied simultaneously in different regions of the visual field. On the other hand, their application is probably not simply successive; T/I does not triple as we go from two letters to six. Pending further research, we can only state that fully parallel operation, at least, is not the rule for spatially separate inputs.

4) On the other hand, the clear-cut results with "QvZ" show that fully parallel operation can be achieved among various shape-recognizers acting on the same input. There can be no doubt that the system examines different features of the stimulus pattern in looking for "Q" than for "Z", but the examination uses no extra time. It follows that different elementary figural properties can be processed simultaneously.

In a projected experiment, we will examine "or" functions of more than two variables, such as "QvPvXvZ". One wonders whether there is an effective upper limit to the number of parallel searches which can be carried on. It seems at least possible that there is no such limit. The question can be referred to a common experience: anyone can scan a crowd to see if it contains a familiar face. Does scanning time increase with the number of recognizable acquaintances we have made in the past?

5) The equivalence of the T/I values for all the negative and mixed functions in the two-letter case seems paradoxical at first. We would have at least expected "-Q" to differ from "-Z". A look at the list themselves explains the paradox, however, and emphasizes the great flexibility of organization which characterizes human pattern recognition. In "-Z", as in "-ZvQ", each item except the critical one contains a "Z".* Since the items are but two letters long, the "Z" in any item has a 50-50 chance of lying exactly underneath the "Z" in the preceding item. The visual effect is the formation of short and long columns of "Zs", shifting haphazardly between the left and the right sides of the list. The subjects commented spontaneously that they handled these lists differently from the others, following these columns with their eyes without heeding the horizontal structure of the items at all. Essentially, they were using different shape-recognizers. The features that distinguish "Z" from other letters in a row are not the same as those which mark the continuity of a column of "Zs". The column-continuity features of "Z" seem to be no different, or at least no more complex, than those for "Q".

This result, more than any other, emphasizes the extraordinary adaptiveness of human information processing. It is unlikely that any artificial device we will know how to make in the near future will be as efficient in taking advantage of the vagaries of its environment.

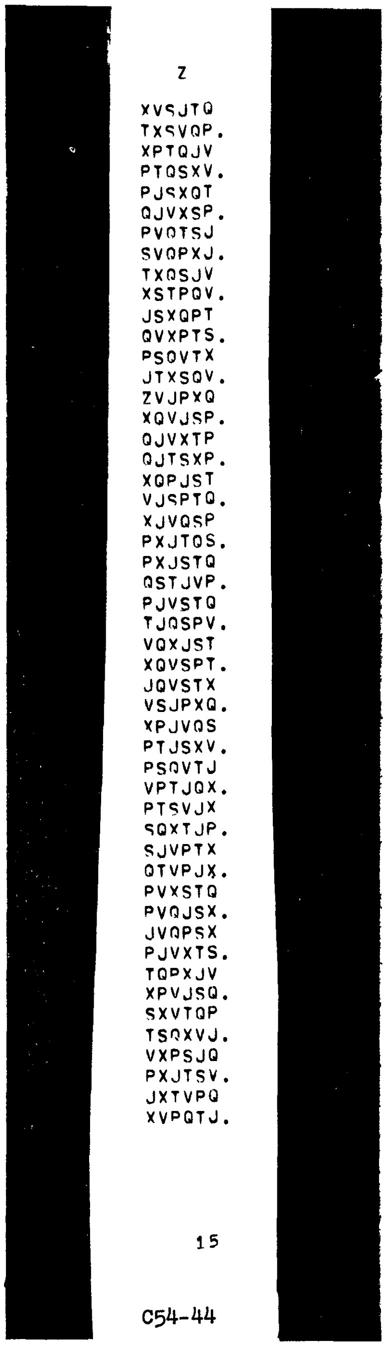
6) The remaining results are not easy to interpret. The T/I for "-QvZ" is much longer than the model we have been using would predict. We would have expected it to be near the value for "-Q".

In summary, we have presented a rough model of human information processes, a method for studying these processes in detail, and some preliminary data obtained by the method. The model is simply a hierarchical organization of specialized subsystems, called "recognizers"**, which effectively abstract certain features from their input. The method consists of measuring the average time needed to process successive items of a list that is being scanned for some particular characteristic. The data suggest that different elementary recognizers have different operating times; that different recognizers can operate simultaneously on the same input; that spatially distinct parts of the input cannot be handled entirely simultaneously; and that the processing hierarchy can be flexibly adjusted to meet the demands and opportunities of the task.

---

*In the case of the function "-ZvQ" the critical item itself may also contain a "Z" (if it contains a "Q"). However, this does not substantially change the present argument, though it suggests that the subjects can check for the presence of a "Q" simultaneously with tracking the columns of "Zs".

** The model is a version of Selfridge's "Pandemonium", in which the subsystems were called demons.



```
    Z

XVSJTQ
TXSVQP.
XPTQJV
PTQSXV.
PJSXQT
QJVXSP.
PVQTSJ
SVQPXJ.
TXQSJV
XSTPQV.
JSXQPT
QVXPTS.
PSQVTX
JTXSQV.
ZVJPXQ
XQVJSP.
QJVXTP
QJTSXP.
XQPJST
VJSPTQ.
XJVQSP
PXJTQS.
PXJSTQ
QSTJVP.
PJVSTQ
TJQSPV.
VQXJST
XQVSPT.
JQVSTX
VSJPXQ.
XPJVQS
PTJSXV.
PSQVTJ
VPTJQX.
PTSVJX
SQXTJP.
SJVPTX
QTVPJX.
PVXSTQ
PVQJSX.
JVQPSX
PJVXTS.
TQPXJV
XPVJSQ.
SXVTQP
TSQXVJ.
VXPSJQ
PXJTSV.
JXTVPQ
XVPQTJ.


      15

C54-44
```

FIG. 1

584
13.4

A-13211 NEISSER Slide McKeon 2-27-61 A-13211

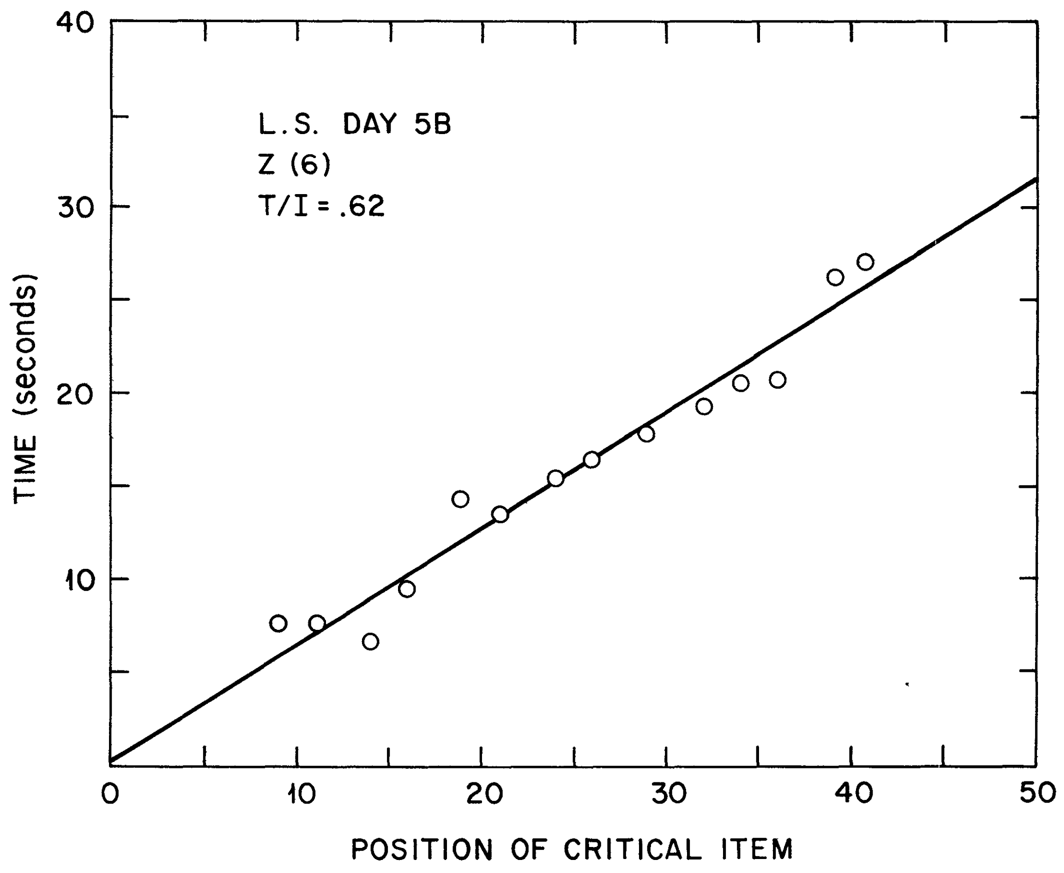


FIG. 2

C54-45

Length:	two letters				six letters			
Subjects:	SS	LS	MA	mean	SS	LS	MA	mean
Function								
Q	15	13	07		20	16	11	
	14	06	06	(10)	14	13	08	(14)
Z	23	31	19		38	62	60	
	26	22	17	(23)	70	49	51	(55)
QvZ	24	22	20		52	86	59	
	25	20	16	(21)	47	44	51	(57)
-Q	34	35	33		36	68	46	
	32	28	28	(32)	30	60	36	(46)
-Z	34	36	40		56	96	73	
	30	34	26	(33)	60	88	52	(71)
-ZvQ	30	39	28		60	96	90	
	32	37	33	(33)	68	98	58	(78)
-QvZ	34	32	28		132	104	75	
	36	47	28	(34)	95	82	66	(92)

Table 1. T/I as a function of experimental conditions and subjects. Replications appear with the second beneath the first. Times are in hundredths of a second. Each mean is based on the six figures to the left of it.

COMPUTER-BASED MANAGEMENT CONTROL

Alan J. Rowe, Manager
Industrial Dynamics Research
Hughes Aircraft Company
Culver City, California

Summary

The use of computers for management controls poses an entirely new set of requirements on the system designers. Tied into automating information processing is the question of an adequate understanding of the control problem itself. For example, measurement or management reporting is often confused with the control process. Although information processing is an integral aspect of a management control system, nonetheless there are other considerations which should be taken into account such as, interdependencies of system components, response characteristics, adaptive capability, decision rules, feedback mechanisms, etc. Furthermore, current methods of measurement are based on historical averages of past performance, whereas what is needed is timely information which reflects actual performance. Finally, the computer, through the use of simulation models, provides the capability of pretesting system designs and the basis for eventual real-time control.

Introduction

With the ever-increasing use of computers in business, there is a growing demand for computer-based management controls which have the capability of operating in real time. Although there are scattered examples of real time control systems such as SAGE, SABRE, and PERT, there is by no means wide-scale use of these systems nor a basic understanding of the control process itself. Rather, the majority of computer based systems can best be described as "mechanization" of existing manual systems. To make effective use of computer technology, therefore, a body of knowledge is needed in management control, feedback theory, information theory, organization theory and the techniques of operations research, management science, computer programming, etc. It is no wonder, therefore, that progress has been slow in achieving truly "designed" computer-based management control systems.

Why Computer-Based Systems

One can hardly question the complexity and dynamics of modern organizations. In view of shorter lead time requirements, increased number and complexity of products, wider geographic distribution of customers, and potentially larger risks in decision making, management can no longer afford the luxury of operating on hunch, intuition, or guesswork. To have effective control, management requires timely information which shows the impact of decisions on the total business, decision criteria which permit rapid response to changing conditions, and organizational design based on information requirements.

When considering the management control problem, one cannot ignore the question of design of the business itself as an economic system. Business systems are run by managers who are responsible for organizing and utilizing the available resources subject to the goals and objectives of owners, and subject to constraints such as capital structure, physical facilities and market status, in order to assure effective operation and survival of the enterprise. Profit maximization can hardly be stated as the single corporate objective; rather, a match of the willingness of ownership to take risks with the inherent capabilities of the business system produces a set of feasible alternatives. Until recently a methodology has not been available to evaluate the impact of alternative strategies in a dynamic environment; however, there are now a number of instances where computer simulation has been used as a research and design tool for this purpose.

Role of the Computer

It is no accident that the computer has assumed a dominant role in business applications of information processing. Likewise it appears highly probable that the computer will play a vital role in the development of real time management control systems; for herein the computer is no longer used merely as a high-speed data processor. Rather, the computer, through simulation models whose behavior corresponds to actual systems, provides a policy laboratory to pre-test new designs of management control systems, as well as the capability of permitting real time decision making. Computer simulation has been used for a long time to solve difficult problems such as evaluation of complex integrals. Simulation has also been used successfully to design manufacturing plants, to develop scheduling decision rules, to plan flight maintenance operations, etc. However, the real power of simulation as a research vehicle to conduct laboratory experimentation has hardly been tapped. Not only can insights be gained by stressing systems beyond normal conditions, but valuable information on the sensitivity and interaction of variables is possible. In testing new decision rules or control theories, simulation provides a systematic means of problem formulation and application of statistically designed experiments. An important attribute of simulation is the capability of demonstrating new system designs by comparing alternatives for management, as well as using the simulation program for training when implementing the new system. Probably the most significant aspect of simulation is the role it can play in projecting expected system behavior which is necessary for real time management control system operation.

Computer Programming Considerations

In the application of computer-based systems, it is evident that there is a need for precise formulation of the computer program. Descriptive statements are not sufficient; rather, quantitative or analytic formulation of problems is required. Furthermore, factors such as kind and size of memory, speed of computation, manner of indexing, and rounding must all be taken into account in computer programming. Methods of filing information, data accumulation, and reporting requirements are also significant aspects of program design. It is often necessary to reformulate a problem to conform to the computer requirements; although this consideration is becoming less important with the availability of large-scale digital computers.

At the outset of a given computer program, a decision must be made whether to have a flexible, general purpose program or one designed for the immediate specific use. Modular programming which treats each section of a program separately provides considerable flexibility at only a small cost in computation time and storage. Probably the most difficult aspect of the problem is the actual coding or programming language. Not only is the coding a time-consuming and difficult process, but the system designer must convey the intent of his work to programmers, thus compounding the problem. Considerable effort is currently being expended in the development of computer languages which can be used more readily by system designers. This, in part, recognizes the fact that as much time is often spent in coding a problem as in the formulation. In this regard, then, considerable work remains to be done to develop computer languages which will simplify the operational instructions for the computer programming task.

Defining Management Controls

In current usage of management controls, measurement is often mistaken for control. To carry out its control function, management utilizes a communication and reporting system; however, control also includes specification of objectives, decision criteria for evaluation of performance, and decision rules for corrective action. Thus, a management control system can be described as a combination of a data processing system, a management information system and a control system. These can be represented schematically as shown in Figure 1.

Data processing is typically carried out at level 2, whereas a management information system includes levels 1, 2 and 3. To achieve control, however, a fourth level is needed which includes decision rules, corrective action, and a feedback mechanism which can change the plans as a function of actual performance. If the corrective action responds in time to change performance during the activity, this can be called a real time control system. If a computer is used for data processing, in addition to this response capability, then we have a computer-based management control system operating in real time.

Achievement of a real time control system, however, is dependent upon both an understanding of the behavioral characteristics of business systems and design principles which help meet desired specifications. Unfortunately, most business behavior is confounded by the decision rules used, so that empirical observations are not sufficient. That is, observed behavior is a result of an underlying phenomena, such as product demand, and the decision rules applied, e.g., pricing. Thus, if the pricing decision rule is changed, a modification in demand might be expected, however, the manner in which demand would change may not be predictable if based on the observed behavior using any one pricing rule. An example of underlying phenomena in a business system is queuing. Where queuing theory is applicable, expected waiting time can be predicted for alternative queue disciplines. In essence, this is a problem of joint variation, where the prediction depends on a knowledge of the behavior of individual variables. Although these problems often cannot be solved analytically, simulation techniques appear to offer a means of establishing these relationships. In addition to the difficulty encountered in describing business behavior, there are few design principles. It is thus obvious that considerable research is still required before optimal system designs can be achieved.

Requirements of Management Control Systems

Although the design of these systems is still, at present, a difficult task, an analysis of requirements can provide useful insights into the problem. In most organizations, control is really a throttling technique which prevents major upheavals rather than providing the means for achieving specified objectives. Thus, for example, budgets are used to prevent excessive expenditures, rather than attempting to find suitable measures of performance. Without such measurement, there cannot be true control in the sense of meeting objectives. Not only are the measurements currently used inadequate, such as historic averages, but they often lead to conflict among divisions of a company. For example, by limiting overtime in one department, a sale could be lost due to late delivery in another department. However, individual managers are measured on their own performance without regard to the enterprise as a whole. Hopefully, computer-based systems with their ability to provide more timely and accurate information can help remedy this situation.

A related problem to that of measurement is an understanding of the functional interdependencies among divisions of a company. By establishing suitable decision criteria, it would be possible to minimize conflicts and avoid suboptimization. Furthermore, decision rules which provide a systematic response to dynamic conditions help to assure desired performance. Where these rules are correctly designed, they relate the basis for plans to the implementation and control requirements. In this respect, the computer may provide the means for pre-testing new programs and their related decision rules to provide assurance of expected performance.

Design of Controls

For purposes of this paper, control is considered directly related to the decisions made in a given system. If we examine the typical decisions made in a business, we find that there are those which affect the entire system; for example, resource allocation, and other decisions that do not have an appreciable effect on the entire system. The former decisions are generally long range in nature, whereas the latter tend to be operational. However, the majority of decisions directly affect a number of other system activities. Because of these inter-dependencies, the decisions are difficult to formulate explicitly. Thus, one of the major advantages of a computer-based system is the capability of continuous updating of interacting variables in the system.

Turning to the question of designing controls, they should be capable of assuring the following response characteristics in a business:

1. Adapting the system to changing conditions.

2. Stabilizing system response under variable demand.

3. Allowing maximum effectiveness of the system rather than imposing restrictive limits.

4. Providing a minimum time lag in correcting deviation in performance commensurate with desired objectives.

5. Implementing real-time control.

Although the possibility of achieving real-time control appears very likely in view of the capability of computers to rapidly process data, the data review frequency should be matched with the response capability of the system. For example, if corrective action requires a minimum of one day, a review frequency of once a minute would be unreasonable. Since business systems often have long delays between measurement and corrective action, the significance of real-time control may differ radically from that used in military control systems.

In business, there are many interacting subsystems, and response depends on factors such as the status of subsystems, speed and amount of their changes and time dependencies. The time dependencies are important in the design of error correcting controls. The response period is dependent both on the magnitude and the rate of change required. Where the response period has to be made extremely short, as in military control systems, computer-based controls are often the only solution. Short response periods, however, may be prohibitive for business systems due to cost. In the design process, cost of control is often overlooked; however, it is a significant factor and should be balanced with operating efficiency.

Another consideration in the design of control systems is the accuracy of error sensing. Where there is loose coupling between system response and the forecast, extreme accuracy is unnecessary. However, where accuracy is required, the forecast can be improved by considering the current state of the system as the base point for predicting future states. Thus, the best estimate of the state of the system at some future time can be obtained by a form of exponential smoothing using both past history and anticipated events.

A related question is the accuracy required in the response mechanism. Since business is stochastic in nature, error correction should hold performance within a specified variance band rather than directed toward an exact objective. Thus, correction is made in two stages. First, a large correction is made, and then only minor adjustments within the variance band, which is similar to quality control techniques.

Since decision rules are used as control mechanisms, they can affect performance by inducing fluctuations in an otherwise stable system. For example, consider a classical inventory problem, where the usage rate is constant and the replenishment rate has a lag imposed by a reorder point rule. The decision rule which leads to periodic replenishment will cause fluctuations in employment if the plant is producing a single product. Thus, labor productivity would fluctuate despite constant demand. This condition might arise due to the physical processing where production could not match consumption. Thus, a better understanding of the interaction of decision rules and physical processes is a prerequisite to the design of effective management control systems.

Control Related to System Characteristics

In the design of management control systems, both variability of system performance and response characteristics should be taken into account. We can anticipate that the control methodology and information requirements for control would change radically for different types of business, if the variability of performance is assumed to increase as a function of the complexity of the business. Variability in the system is a measure of the error in predicting performance. Furthermore, it is assumed that the quantity of information required for control varies inversely with the stability of the system. Thus, a system which exhibits considerable variability would require a larger amount of information to maintain a given level of control. Of course, it is assumed that variability is caused by underlying phenomena rather than induced fluctuations resulting from the control system employed.

Another problem is that of local vs. total system control. Stated another way, the question is, what is the optimum combination of subsystem controls where there is interdependence? One approach to this problem is the use of quadratic programming to find the optimum combination defined in terms of expected values, variance and co-variance estimates. Thus, the control system can be designed to achieve a given expected value with a specified associated risk. Since survival of the enterprise is an important management objective, this manner of formulating the problem of joint optimization should be useful in control system design.

Organizational Considerations

Since top management is primarily concerned with long-range decisions, there is a loose coupling with the actual business processes. Thus, the amount of aggregation and frequency of information transmission should be appropriately designed into the measurement system. Rather than continuous reporting of system status to top management, periodic reports are generally sufficient. However, a computer-based system would provide the capability of random access to updated information on detail system status as the need arose. In this sense, then, top management would have real-time controls.

A related problem to reporting of system status is the communication network in an organization. In current management control systems, measurement of performance is reported successively upward with each level summarizing information to the next higher echelon, until it reaches the top. At each level, interpretation, summarization and biasing occurs. Although the levels act as filters in one sense, they can also introduce distortion in another sense. Furthermore, at no time does anyone but the top executive level have access to all of the information in the system.

In a like manner, the design of the decision network poses a number of formidable problems. Current organization structures lead to a cascading effect of decisions as a result of interpretation at each level and generally differing measures of performance. In addition to the cascade effect among levels, there are the conflicting objectives at any given level. Thus, there is need for a design methodology which treats the problem as a whole rather than the fragmented approach prevalent in industry today.

Contrast this with the concept of a systems staff reporting to the top executive which acts as the summarizing mechanism using the computer for rapid, random access to detail operating information. In this kind of application the computer would have direct access to all system information and have a large percent reported automatically. Not only is the information on system performance more timely and accurate, but the system staff, as a team, has an overview of the entire operation. Thus, we can anticipate that the organization design in computer-based control systems will be more closely matched with the information requirements of the system.

Conclusion

Although there are many formidable problems still to be solved, the advent of computer-based management control systems for business systems is becoming a reality. However, to avoid the "mechanization" approach, design principles should be applied which consider the problem from the total system viewpoint. Although there may be organizational changes accompanying a computer-based system, the net effect should be better decision making and control due to more timely and accurate information, as well as the use of suitable decision criteria and decision rules. The manager, in a real time computer-based system, can be expected to perform more effectively since the system will have been "designed" rather than growing like Topsy.

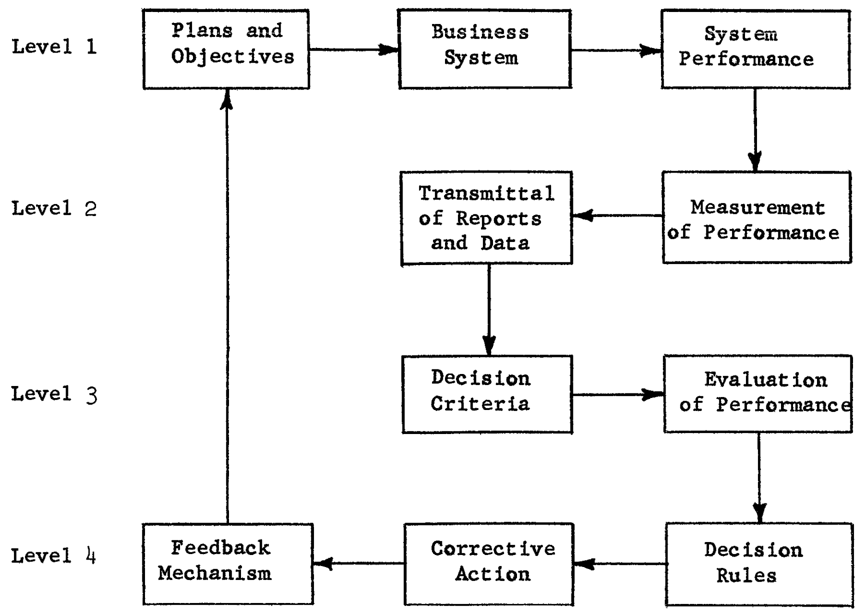

Level 1
Level 2
Level 3
Level 4
Plans and Objectives
Business System
System Performance
Measurement of Performance
Transmittal of Reports and Data
Decision Criteria
Evaluation of Performance
Decision Rules
Corrective Action
Feedback Mechanism

Figure 1

AMERICAN AIRLINES' "SABRE" ELECTRONIC RESERVATIONS SYSTEM

W. R. Plugge, American Airlines, New York, New York
M. N. Perry, American Airlines, New York, New York

Summary

The American Airlines Sabre System, a joint development of American Airlines and IBM, is a major step into the field of total data processing. This system is designed to solve the problems confronting the airlines in passenger sales, seat inventory control, and maintenance and retrieval of passenger records.

After six years of joint effort, a complex of programs and hardware has been developed which will be the largest commercial data processing system in existence. At the heart of this duplexed system are two IBM 7090's. At the extremeties are agent consoles built to American Airlines specifications. This system will automate all of the daily reservations processes with the exception of the vital agent-customer contact.

The Sabre System will give, in addition to the obvious customer advantages, the availability of current, detailed and summarized data for the use of American Airlines' management in their constant endeavor to improve passenger service.

Purpose Of The Sabre System

Progress in the air since 1930 from the DC-3 to the DC-7 was matched with similar, but less dramatic advances in our Reservations offices. The dynamic and revolutionary burst into the Jet Age with the Boeing 707, DC-8 and now the Convair 880 and 990 has presented the airline industry with new problems in the Reservations function because of the jet aircraft size and speed. These airplanes which can carry up to 150 passengers can depart an airport and in some instances arrive at the next downline city before our present day reservations systems has adjusted the passenger inventory.

In the year 1960, American Airlines carried 8,615,000 passengers. In terms of reservations' phone transactions, this figure must be multiplied by a factor of 3 or a total of 26,000,000. The Sabre System being installed for American Airlines in 1962 will assist us in processing 85,000 daily telephone calls -- 30,000 daily requests for fare quotations -- 40,000 daily passenger reservations -- 30,000 daily queries to and from other airlines and 20,000 daily ticket sales. All of this processing for most individual interrogations will be handled in less than three seconds.

Sabre's main purpose is to carry out on a nation-wide basis, the functions associated with the sale and control of air transportation from the customer's first call for information to his arrival at his final destination. To achieve this purpose, Sabre will perform a large number of different functions which can be grouped into three main areas; Passenger Sales, Reservations Record Service and Management Reporting.

Passenger Sales

The primary role of reservations sales agents and ticket sales agents is to sell American Airlines' space and provide the quality of customer service which will encourage passengers always to turn to American for their air travel needs. The Sabre System was designed to help the agent provide this kind of customer service with increased speed and accuracy.

Reservations Record Service

The electronic processing center will perform a number of record service functions, which also support the sales efforts of agents in the Field and promote efficient flight loading. These functions can be grouped into three main areas; information maintenance, distribution of schedules and operating changes, and teletype message handling.

Management Reporting

In addition to Sabre's Passenger Sales and Reservations Record Service, the system will provide as an important by-product, management control information. The electronic processing center automatically can and will prepare several management reports at given intervals.

Over six years of joint American Airlines - IBM development effort has been devoted to this system, which has now emerged as a comprehensive reservations and control device, designed both to serve the air traveler more swiftly and effectively and to offer American Airlines increased sales effectiveness and better utilization of passenger space. The Sabre System will contain two duplexed IBM 7090 computers, disk and drum storage devices of advanced design and much greater capacity than any now in use, and a specially designed on-line data communications network with remoted input - output devices -- all on a scale never before seen in a commercial application.

The system will maintain a complete and up-to-date inventory of passengers booked and seats available. It will enable reservations agents to confirm, cancel or alter reservations and determine seat availability in a matter of seconds. And it will enable reservations agents to obtain the passengers name, record and all pertinent data from the storage devices in a matter of seconds. One of the most important and unique aspects of the Sabre System is that it will operate on current information and will be involved in the control and execution of current transactions. The accumulation of historical data will be a secondary function. The Sabre System is not a record keeping device, but it is an operating real time system in operation 24 hours a day, 365 days a year.

Why We Need Electronic Computers For Reservations Control

The airline industry, including American Airlines, cannot afford to operate $5\frac{1}{2}$ million dollar airplanes without reservations. The balancing of equipment, government regulations to adhere to scheduled operations, food services, and last but not least, our competition dictates that to run a profitable airline, you have to have reservations.

The Sabre System represents a major step in what has been an evolutionary process in our reservations function. Twenty years ago, we installed availability boards in our larger offices. As volume increased, we soon found the availability boards crowded, more and more flights were added and our employees had to sit further and further away from the board. It became obvious that some system of providing information directly to the agent position was necessary. We began exploring means of replacing the manual effort for determining seat availability with faster, more accurate mechanical methods. In 1944 we developed the idea or concept of a mechanical system which would keep our sales agents informed of seat availability on various flights. We found an equipment manufacturer - Teleregister - to develop the system. By 1946, such a system called the Reservisor was installed in our Boston Reservations Office. The original Reservisor was a combination of input and output units which interrogated a central source and reflected the availability status on a given flight - "OPEN" for sale, or "CLOSED." It did not allow for the automatic register of "SELL" and "CANCEL" transaction, but it was a milestone in reservations' history, because it was the first time any airline had adapted current electronic discoveries to reservations handling.

Further research and development led to a larger and more complex Reservisor called the Magnetronic Reservisor which was installed at LaGuardia in 1952 to handle our mounting tide of passengers. The significant advancement found in the Electronic Reservisor was the introduction of an arithmetic ability and a memory drum to the system, which allowed sales agents in the New York area not only to determine seat availability, but also to automatically "SELL" or "CANCEL" seats recorded on the drum.

While the Reservisor's devices brought significant improvements to the reservations handling function, we saw the need for further improvements in the overall reservations area. Several problems were arising because with a growing volume of passengers, we faced increasing difficulty in keeping seat inventories in conformity with the reservations records. There was no direct or automatic link between the Reservisor seat inventory system and the passenger and reservations record, which were handled by manual methods. Serious error problems were arising in the handling of reconfirmations and wait lists, which were further aggravated by delays in finding the errors. As a result of these problems, customer service and efficient aircraft loading were suffering. For example, we often found the inventory count of passengers booked would not agree with the reservations records and this resulted in either oversales or undersales, i.e., either more seats were sold than were available under the original flight schedule or seats were not sold to requesting customers even though in actuality they were available. Undersales also arose because cancellations did not result in a timely reopening of the affected flight. In addition, in order to allow for sales already in the communications "pipeline" but not reflected on the records, a certain number of seats were held open as a "cushion." This "cushion" often was not fully absorbed by sales in the "pipeline" and was not reopened in time to make the seats available to customers on the waiting list.

We also were reaching a point of diminishing returns in terms of reservations manpower. As the passenger volume grew, more and more reservations manpower per passenger boarded was required for communications and record keeping. This resulted in a cost ratio trend that was beginning to take alarming unfavorable proportions.

Foreseeing the affect of further increases in passenger volume upon the efficiency and accuracy of the present reservations handling system, we began exploring with IBM the possibility of further improving the reservations operations. In 1953 American Airlines and IBM formed a joint engineering -- Product Planning Project. Staffed by both American Airlines and IBM working in close cooperation, this joint team performed a thorough detailed overall system analysis in the sales and reservations area, to determine the characteristics and volumes of the existing operation and to develop a program to solve the overall problem. The approach was not one of what equipment was available on the shelf to do the job, but rather what equipment was needed with the thought being that whatever was needed would be developed from the ground up.

Benefits Of The Sabre System

In all three of the major groups of functions performed -- Passenger Sales, Reservations Record Service and Management Reporting -- substantial improvements will be realized.

The customer himself will be a major beneficiary of the system. His request will be processed more speedily and accurately with timely information that reflects the actual status of seats available. This superior service, we hope will attract more customers and therefore more revenue to American Airlines. This system will also increase agent productivity, which should result in improved sales.

Our aircraft will be more efficiently and fully loaded, since cancellations, "no-shows," and waiting lists will be processed immediately and accurately. Also, with more timely and accurate information it will not be necessary to maintain a "cushion" of unsold seats to handle sales in the communications "pipeline," or to cover clerical errors. American Airlines' reservation agents throughout the United States will have an agent set which is connected to the computer via the Sabre communications network. With this, they will be able to request and obtain a reply on seat availability instantaneously on any flight on the American Airlines System. Management direction, both of an operating and planning nature, will be based on more accurate and timely data. This should allow for a more effective and better informed control and evaluation of our airline operations.

Sabre's benefits assume a great importance when viewed against the background of the airline industry during the present decade. The costs of both flight crews and equipment in the Jet Age have increased and, therefore, placed a high premium on the efficient utilization or loading of equipment. An empty seat is much more expensive than it used to be from many standpoints. Also, the growth in passenger volumes and the increased speed of aircraft have produced a need for processing a greater volume of data, faster and more accurately. The growth of competitive conditions in the industry accentuate these needs for more timely and accurate information and control.

Operating Characteristics Of The Sabre System

The Sabre System will consist of three main elements: the Electronic Reservations Processing Center in the New York area, the Agents Sets in the field offices, and the Communications Network. The processing center is completely duplexed. The reliability requirements are such that, if necessary, the entire system might be duplexed.

The Electronic Reservations Processing Center will contain all of the system's electronic storage facilities, and will perform logical, computational, and decision-making functions for the system. An important aspect of the center is that all of the memory is randomly accessible. This is one of the features which allow the simultaneous processing of many requests.

At the heart of the electronic processing center are two IBM 7090's, each having a 32,000 word memory. The memory of the computer actually performing the reservations processing will contain the "control program," the messages actually being processed, and operational programs to perform those functions which have been requested. The "control program" has the capabilities of a conventional execution program; but, in addition, will perform all allocation of core temporaries on request, and will allocate space for programs when needed. Due to the magnitude of programs required (well over 100,000 words), only those programs actually in use can be retained in core. Using a hardware relocation device, programs will be shuttled into core each time they are used. The "control program" also facilitates multiprogramming which will be carried on to a degree never before attempted. As many as 30 programs will be in progress at any given time during the peak load period of the day. Each program will proceed until it is required to "wait" for an external reference, at which time, another program will be initiated or allowed to proceed. By this device, each request will be processed in a minimum of elapsed time, making this system truly "Real Time."

In addition to the computer's internal core memory, the electronic processing center will contain magnetic drums and large-capacity magnetic disk files which will act as the system's principal storage media.

The magnetic drums will have capacity for 7.2 million characters and will contain:

1. The inventory of the number of seats sold and remaining for sale on each American Airlines flight.
2. Current flight and schedule information.
3. An area for each of the 1100 agent sets using the Sabre System where requests and messages will be assembled.
4. The more than 100,000 words of programs.

The magnetic disk files will have capacity for over half a billion characters and will contain:

1. Current passenger reservation records.
2. The indices to facilitate retrieval of passenger records.
3. Duplicate copies of all information stored on the magnetic drums.

There will be several buffer units which will enable the computer and the many input/output devices to communicate with each other. These buffer units have the logical ability to schedule, control, and assemble input and output data between the computer and the magnetic drums, the magnetic disk files, the communications lines to the Agent Sets, and other input/output equipment located in the processing center.

In addition to the random access storage devices mentioned above, there will be a large library of magnetic tapes which contain historical information.

Agent Sets will be located at more than 50 cities throughout the United States and are used by American Airlines' Reservations agents to communicate with the processing center in the New York area. To facilitate the use of these sets, each agent will have a file of Air Information Cards, which are pre-coded, machine-sensitive cards showing information relating to all American Airlines flights and certain flights of other airlines.

Each agent set will consist of three principal elements: an Air

Information Device, a Director Console of rapid action pushbuttons, and an input/output typewriter. Air Information Cards are inserted in the Air Information Device, which automatically senses the code punched in the card for transmission to the computer. Row and column pushbuttons on the Air Information Device are depressed to designate a service preprinted on the Air Information Card, and buttons on the Director Console are depressed to indicate the date on which the service is requested, the number of passengers involved, and the type of action required. The computer's response to a request using an Air Information Card will be displayed to the agent by lights on the Air Information Device or on the input/output typewriter. The input/output typewriter is also used by the agent to enter information of a variable nature, such as passenger name, phone number, etc.

The Communications Network, which is similar in operation to an AT&T automatic switching teletype circuit, will consist of over 10,000 miles of telephone lines and switching devices which are required at each of the more than 110 physical locations at which Agent Sets are installed. These switching devices called Mulcoms (Multiplexor-Communications) accomplish the sharing of each of the 9 or 10 separate long telephone lines radiating from the processing center in the New York area. All outgoing messages transmitted from the processing center are monitored by each Mulcom on the line and sent only to the appropriate Agent Set. In the other direction, when an agent using an Agent Set initiates transmission, it is buffered in a logic gate until it is time for its associated Mulcom to use the telephone line. Under this structure, rate of input can be controlled by the computer in the processing center. Periodically (at least every half second), the computer instructs the farthest Mulcom to begin sending. After this Mulcom has transmitted all waiting messages, it instructs the next farthest Mulcom to "go ahead." This process continues until all Mulcoms on the line have transmitted all waiting messages.

Following is a typical example of the use of the equipment which has been described. Let us suppose that a customer telephones the American Airlines' reservations desk in Washington and wishes to reserve a "seat on a flight to Chicago, leaving Washington sometime tomorrow." The agent will select an Air Information Card marked "Chicago" from the file before him. For agents in Washington, this card lists all relevant information for American Airlines' flights between Washington and Chicago. The agent inserts this card into the Air Information Device on his Agent Set, where it will remain in sight throughout the remainder of the transaction. By pushing buttons on the Director Console, the agent records the number of seats required and the date for the trip. After discussing the desired flight with the customer, the agent depresses the row and column pushbuttons on the Air Information Device to designate the flight most convenient for the customer. He then presses the "Sell" button which sends all of the information which he has entered through a Mulcom and thence to the computer in the processing center. If, in scanning the inventory records, the computer determines that the desired seats are available, a message is sent to the agent (printed on his typewriter) giving him information concerning the flight and advising him that the seat has been reserved. If, on the other hand, the seat had not been available for sale, a display of lights on the Air Information Device would inform the agent on which flights he could reserve seats. Assuming that the seats were available, the agent would ask for the customer's name, home and business phone numbers, the name of the person making the reservation if not the customer, and any special requirements the customer might have, such as car rental, wheelchair, special diet, etc. The agent enters all of this information, properly identified, via the input/output typewriter, into the computer. At the end of the transaction, the agent depresses the "end transaction" button. If all of the necessary information has been entered, the computer will send a confirming message to the agent and will construct, file, and index a passenger record for future reference. If, at some future time, whether 1 second or 1 year later, the customer wishes to revise his reservation, this record can be retrieved and used as the point of departure for his revised reservation.

The foregoing example showed the most normal use of the Sabre System. Some of its other vital, if less-used, capabilities are briefly outlined below.

Teletype Communications

Requests for continuing space on other airlines, car rentals, tours and hotels, and taxi service, may be entered directly into the agent's set. If a request is to be executed by American Airlines, it will be transmitted by Sabre to an office in the appropriate city. If it must be executed by another airline, a teletype message will be transmitted to the proper point on that airline. Airlines which are permitted to sell space on American Airlines will notify American of each sale by sending a teletype message. When a designated inventory level is reached, Sabre must generate and transmit stop sales messages via teletypes. Naturally, Sabre must be prepared to accept and interpret each type of message that it generates. An interesting and unusual example of worldwide industry cooperation is provided in the development by the Air Transport Association of America and the International Air Transport Association of a machineable interline message format which specifies rigidly the form and control of each such message.

Waitlists

Waitlists are created and processed for all flights. If a cancellation occurs, the first passenger on the waitlist is "confirmed" and a confirmation message sent to him via an American Airlines agent.

Flight Forecasts

Flight forecasts are maintained for all flights which have not departed. If the forecast is "not normal," this forecast information is furnished automatically each time a reservation is made on this flight. Flight progress is maintained on each flight after it departs. This information is furnished to the agent on request to facilitate answering inquiries concerning details of the flight. Flight forecast and progress information is entered into Sabre via agent sets.

Ticketing Arrangements

Ticketing arrangements are entered as part of each passenger record. Ticket pick-up time limits are tabulated and periodically, the list is scanned to determine if any of the limits have been expired. If a time limit has expired, the reservation is automatically cancelled in most instances. In a few cases of especially complicated itineraries, the passenger must be contacted. Information is furnished by Sabre to facilitate this contact.

Flight Manifests and Passenger Name Lists

Flight manifests and passenger name lists are furnished for each flight and upon request. These name lists are used to check against boarding passengers. It greatly facilitates detecting "no shows."

Processing of Schedule Changes and Extra Sections

Processing of Schedule changes and extra sections will result in automatic generation of lists of affected passengers and how they may be contacted.

Historical Records

Historical records are retained (on magnetic tape) to allow investigation of transactions for some months past as prescribed by law.

Various Operating Statistics

Various operating statistics will be gathered to allow improvement of the System, to predict saturation of the present equipment, and provide data for management control.

Other On-Line Applications Being Considered

At the outset, the Sabre System will perform the Reservations functions as described in this paper. In the final analysis, however, many other on-line applications will be put on the Sabre System.

There are several ways that a company can approach the task of developing an integrated real time inventory control and information retrieval system.

One approach is to do a total system study and establish broad company objectives which then are translated into system requirements. From these requirements a system is designed and implemented in one step on a company-wide basis.

Another approach is to select electronic equipment which has expansion capability as the basic components of the system, to implement an important company function and to concurrently study other applications for later application. This is the approach being followed by American Airlines in Sabre. The Sabre System uses standard IBM 7090's as the heart of the system. The reservations function is large and important enough to justify the installation of the minimum data processing and nation-wide high speed communications network.

Our immediate objectives are to get the system paying for itself as soon as possible, to develop our own experienced computer applications staff and at the same time learn more about the problems and capabilities of real time data processing.

Concurrently, we have embarked upon a study and planning effort to meet the following objectives:

1. Determine the expansion capabilities of the Sabre System.
2. Study potential applications.
3. Rank an application in order of its operational need or return on investment.

The results of this study will be a three to five year plan for orderly Sabre growth.

Some of the areas which may benefit in the future from real time data processing are Customer Services, Flight Operations and Maintenance. I shall discuss a few of these briefly.

Customer Services

Ticketing. Such functions as seat confirmation and automatic fare computation and printing could be accomplished.

Air Cargo. Freight inventory control and informational retrieval, optimum space allocation, tariff calculation and billing and freight forwarding messages are feasible.

Lost and Found Baggage. An up-to-date inventory of misguided baggage coupled with a means of rapidly locating it and forwarding it to its owner could be accomplished.

Flight Operations

Flight Dispatch. By providing current aircraft status, it would be possible for Sabre to assist in aircraft rescheduling and re-routing, fuel load calculation, flight plans and clearance requests. In an area such as this, the planning would be done off-line by operational personnel. The Sabre System would provide "how goes it" information and it is conceivable that sometime in the future it would be possible to simulate the effect of a given operational decision so operational personnel could select optimum solution to a multi-factor problem. An example of this is the problem of getting an airline back on a normal schedule after operations have been curtailed over a large portion of the country by a three day snow storm.

At the present time, American Airlines is conducting Ticketing and Air Cargo Feasibility Studies. In the very near future, we will be investigating crew scheduling and communications.

Management Reports And Control By Exception

The Sabre System while basically devoted to the processing of reservations will be extremely useful in supplying Management with an abundant amount of information on day-to-day operations. In addition, Sabre will be able to pin-point critical values from the vast amount of data and "flag" such information to Management. This latter concept known as "control by exception," is especially effective with Sabre because of the tremendous processing capabilities inherent. Management Reports are divided into 3 basic areas:

1. Information that must be

supplied to Sabre Management.

2. Information which is required within the framework of American Airlines, i.e., to divisions of the company outside of Sabre.

3. Information which must be given to agencies outside of the structure of American Airlines.

Information for Sabre Management

In order to evaluate whether the Sabre System is meeting the response time of interrogations entered by the sales agents, several measurements of critical variables will be obtained from Sabre.

One critical variable that Sabre Management is concerned with is saturation of the system. For increased passenger boardings per period of time, additional passenger name records will be composed and stored in the disk files. Consequently, a point in time could occur where the disk files are full. Such a situation will be made known to Sabre Management via an unsolicited computer response. If such a contingency continually arises, Sabre Management may decide to add additional disk files to the system to alleviate the saturation condition.

A second critical variable is the "line loading." An analysis of communication loading in the Sabre System may reveal that the elapsed time from sales agent to computer and return is continually increasing. The computer will be able to measure the response time and print the values per period of time. If the response times are too high, Management may add facilities or reroute interrogations to the computer. If too low, economies can be achieved through consolidating facilities.

Several other critical variables which can be obtained from Sabre for Management decision include:

1. Measure of extent to which Sabre uses teletype equipment, i.e., volume of teletype messages entering and leaving the computer.

2. Measure of the extent of usage of programming routines.

3. Measure of the extent of reference to disk and drum files.

It is intended, therefore, to utilize Sabre to its fullest capabilities in rendering information which heretofore was either unobtainable or laboriously calculated.

Information for other Departments of American Airlines

Sabre will be able to retrieve on command, past-date records from the historical files to satisfy the needs of Management outside of Sabre. The basic purpose of such reports is to illustrate past performances and to predict future goals.

1. These reports may be regularly recurring reports, with specific information occurring at stated intervals, or

2. Reports which may be useful over designated lengths of time, or

3. "One-time" analyses of special interest.

For (1), Sabre will be able to compute daily load factors (ratio of passengers boarded to total seats authorized) per flight for each airport. Under present procedures, such information is not immediately available in total for all airports. Sabre will be able to pin-point those flight legs whose load factors are outside of a certain range. For a low load factor, a sales program would be instituted. A high load factor may be an indication that extra sections should be brought into service for frequently travelled flight legs.

For (2), Management will be interested in such information as:

a. A report of production for the past month which comprises a breakdown of sales by station, activity, sales account and sales agent.

b. A report on reservation-making habits of customers for a particular period of time, as an aid to the sales program.

c. A report on source of business; for example, percentage of business derived from other airlines, commuters, conventions, etc.

d. A report on passengers preference for certain types of meals.

For (3), Management will be interested in such information as:

a. Effect on bookings due to specific advertising campaigns.

b. Measure of extent to which American Airlines is requested to make reservations for passengers on other airlines.

It will be Sabre's purpose to supply a variety of reports to Management instantaneously. (See attachment for examples of Management Reports.)

Information for Agencies Outside of American Airlines

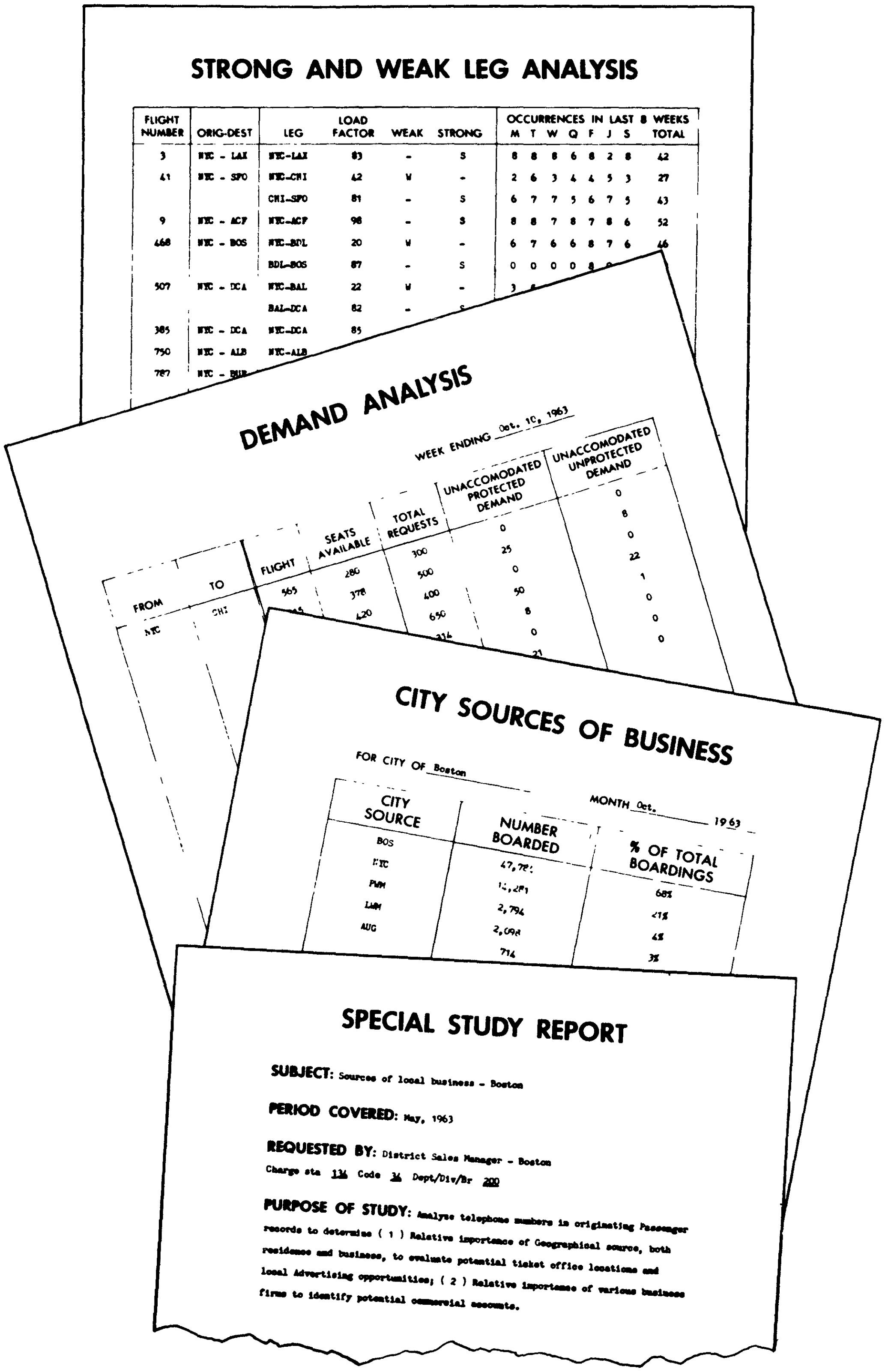
In general, information that is required by organizations outside of the company will not differ in any major respect from that required by the company itself. Such information will be limited to the following outside agencies.

1. The Air Transport Association (ATA)
2. The International Air Transport Association (IATA)
3. The Air Traffic Conference (ATC)
4. The Civil Aeronautics Board (CAB)
5. The Federal Aviation Agency (FAA)

Civil Air regulation requires that airlines retain certain information for a period of 90 days.

a. All Passenger Name Records made for a day including a cancelled reservation or waitlisted passenger.
b. The corrected Inventory Record (record of seats sold for each flight) for the day's flights.
c. Flight information (data governing whether a flight will take off or not) and flight progress information for all flights.

This information will be made available for examination on request. It will be conveniently and efficiently supplied by Sabre from the historical records which are purged from the master files of passenger name records at the end of each day.



STRONG AND WEAK LEG ANALYSIS

FLIGHT NUMBER	ORIG-DEST	LEG	LOAD FACTOR	WEAK	STRONG	OCCURRENCES IN LAST 8 WEEKS M	T	W	Q	F	J	S	TOTAL
3	NYC - LAX	NYC-LAX	83	-	S	8	8	8	6	8	2	8	42
41	NYC - SFO	NYC-CHI	42	W	-	2	6	3	4	4	5	3	27
		CHI-SFO	81	-	S	6	7	7	5	6	7	5	43
9	NYC - ACY	NYC-ACY	98	-	S	8	8	7	8	7	8	6	52
468	NYC - BOS	NYC-BDL	20	W	-	6	7	6	6	8	7	6	46
		BDL-BOS	87	-	S	0	0	0	0	8			
507	NYC - DCA	NYC-BAL	22	W	-	3							
		BAL-DCA	82	-									
385	NYC - DCA	NYC-DCA	85										
750	NYC - ALB	NYC-ALB											
787	NYC - BUR												

DEMAND ANALYSIS

WEEK ENDING Oct. 10, 1963

FROM	TO	FLIGHT	SEATS AVAILABLE	TOTAL REQUESTS	UNACCOMODATED PROTECTED DEMAND	UNACCOMODATED UNPROTECTED DEMAND
NYC	CHI	565	280	300	0	0
			378	500	25	8
			420	400	0	0
				650	50	22
				314	8	1
					0	0
					21	0
						0

CITY SOURCES OF BUSINESS

FOR CITY OF Boston MONTH Oct. 1963

CITY SOURCE	NUMBER BOARDED	% OF TOTAL BOARDINGS
BOS	47,78[illegible]	68%
NYC	14,281	21%
PWM	2,794	4%
LWM	2,098	3%
AUG	714	

SPECIAL STUDY REPORT

SUBJECT: Sources of local business - Boston

PERIOD COVERED: May, 1963

REQUESTED BY: District Sales Manager - Boston

Charge sta 134 Code 34 Dept/Div/Br 200

PURPOSE OF STUDY: Analyze telephone numbers in originating Passenger records to determine (1) Relative importance of Geographical source, both residence and business, to evaluate potential ticket office locations and local Advertising opportunities; (2) Relative importance of various business firms to identify potential commercial accounts.

Examples of Management Reports

REAL-TIME MANAGEMENT CONTROL AT THE HUGHES AIRCRAFT COMPANY

Donald R. Pardee
Industrial Dynamics, G.O.
Hughes Aircraft Company
Culver City, California

Summary

We have reached an impasse as data processing professionals unless we use a new approach in meeting management information requirements. Our old approach - such as batch processing, maintenance of voluminous card and tape files, and rigidly structured reports - can no longer do the job. We must continually recognize the implication and impact of the dynamic nature of the industrial environment. New products, more acute competition, shifting skills, changing customers and reduced lead times place staggering demands on operating managers for faster decisions and more accurate answers. In addition, these rapidly changing conditions make it impossible for management to predict, with any preciseness, what future information needs will develop. What they do know - and demand - is that our information systems must provide more visibility and responsiveness. These systems inherently must be more flexible and contain more economy of operation than traditional techniques afford. At the Hughes Aircraft Company, we are meeting these requirements through the growing utilization of source data recording, large random access file processing, and high character-rate tape units. In addition, we make the greatest possible use of programming aids such as automatic coding routines and report generators. Our Finished Goods and Shipping information system exemplifies the use of these tools in our current approach to real-time control.

Management Information Requirements

As noted recently in the Wall Street Journal management disenchantment with data processing has set in - and will continue to spread - unless we recognize our present shortcomings[1]. We must adopt totally new methods - now - to satisfy today's management information requirements. These requirements have been postulated and expounded upon ad nauseam. Yet, by their very nature these requirements will continue to serve as a criteria and a measure of our successes and failures. It does not seem inappropriate, therefore, that we should review them again.

Quality Improvement of Data

One authority defines information as ".... knowledge derived from the organization and analysis of data" and data processing as ".... the conversion of data into information".[2] Obviously, the value of any information so produced will be a function of the accuracy and timeliness of the data from which it was derived. All too frequently, our old techniques of converting raw data to machine language formats - transmittal preparation, key punching and verifying - have defeated our objective of obtaining current, valid data.

Flow Time Reduction

In the past - and I suspect for sometime into the future - we have been and will be criticized for using our computing power for the sole purpose of producing documented hindsight. Far too great a portion of our data processing reports show where we have been, rather than where we are and where we are going.

Much of this is attributable to batch processing transactions and changes - and our voluminous card and tape files seem to preclude any different approach if economy of operation is to be considered. I think it is fairly obvious that when we batch process for file maintenance and reporting, we are updating files with events that occurred, on the average, one-half the batch-period-interval ago. In short, many of our monthly financial reports reflect important financial transactions that took place 2-1/2 weeks before! This kind of performance will no longer do. If electronic data processing is considered a management science - and I believe it is - our main goal, according to Peter Drucker, ".... must be to enable business to take the right risk". "Indeed", he goes on to say, "it must be to enable business to take greater risks - by providing knowledge and understanding....and by measuring results against expectations thereby providing means for early correction of wrong or inadequate decisions".[3] The flow time between data inception and information reporting can be and must be reduced. As one of our managers has put it, he wants visibility, not perspective; information, not history!

Responsiveness and Flexibility

As data processing people, it seems to me that all too frequently we attempt to ignore the dynamic nature of the industrial environment in which we work. Hardly a day goes by when our companies and management do not experience a change in product or a change in competition, a change in customers or a change in regulatory specifications.

The American Management Association puts it this way, "We live in an ever-changing world. Everything about us - and we ourselves - change to a greater or lesser degree. So far as the business situation is concerned, this state of constant flux introduces two very real complications into our calculations. First, we find as we look into the future that the number of possibilities to be considered increases at an astronomical rate. Second, the further ahead we attempt to plan, the greater the uncertainty factor, which also operates to increase the number of alternative decisions we must consider. On the whole, flexibility and adaptability are urgently needed to adjust quickly to shifting conditions....".[4]

In the face of this situation, data processing must become increasingly more responsive to implementing requests for new applications. It is imperative, I think, that we reduce system's design and programming lead time. In this sense, it will always remain incongruous for us to talk, in the same sentence, of machine processing in terms of micro-seconds and programming in terms of months!

As for flexibility, the difficulty and expense of changing report content, format and sequence has made it mandatory for us to establish controls on change requests. Then, as changes increase in volume and frequency, subtly but inexorably, our controls become impediments designed to discourage change. Carried to extremes, our relative inflexibility leads us to creating reports, we find to our utter amazement, that are neither used nor read.

Hughes Aircraft Company Approach

Automatic Data Collection

At Hughes Aircraft Company, the need to improve the accuracy and timeliness of input data has been recognized as a chronic problem. The multiple operations we required in the preparation of such data by conventional methods permitted the introduction of human error - as well as time delays - at every step. This input gap - between the recording of information at its source and its later processing - has been bridged by the introduction of "Automatic Data Collection" equipment. We are using this equipment currently for:

1. Labor Attendance Recording
2. Labor Distribution Recording
3. Order Location Recording
4. Quality Control Data Recording
5. Inventory Control

In terms of pure economics, the introduction of this equipment reduced clerical costs through the elimination of many handwritten documents; it obviously reduced key punching and verification costs as well as reducing control and audit requirements.

Large Random Access Files

The storage of data on numerous decks of punched cards necessitates multiple machine runs, a great deal of lost time in having to pass the same decks through the system many times for such operations as sorting, merging, calculation and listing. On the other hand, magnetic tape creates difficulties in interrogation of information contained in the tape files and would, in cases in which these interrogations had to be answered quickly, require voluminous print-outs or frequent computer interruptions for tape file searches. Our problem, then, was to develop a system that could produce timely working paper and reports - at a price we could afford. The solution we chose was mass random access storage. Subsequently, we installed a dual-processor, 40 million character storage RAMAC system to compliment our 705 and 1401 computer systems. For the first time, we have the ability to maintain a number of massive files as frequently as four times per day! Having divorced ourselves from batch-process frequencies, we now generate working documents such as work orders, priority reports and delinquency reports on a meaningful basis.

Report Generators

In our efforts, at Hughes Aircraft Company, to increase data processing flexibility and responsiveness, we have and will continue to rely heavily on automatic programming and report generators. In this regard, we have programmed our own report generator, for a tape system 1401, which utilizes control cards specifying format and control total fields, but does not require computer assembly time. Only from extensive use of these "soft-ware" packages can we get the most for our programming dollars - more programs, in less time, with less effort.

Finished Goods and Shipping Control

A brief review of an inventory control system in operation at one division of Hughes Aircraft Company can serve to illustrate an integrated application of real-time source data recording, large random access file processing and report generators.

The Finished Goods and Shipping inventories may be said to represent the culmination of the production and supporting departments' efforts. The products in these inventories - such as end unit "black boxes", shippable spare subassemblies and modification kits - represent a considerable financial investment. Their anticipated shipment is a major consideration in determining realizable income. Of equal importance, these products must be shipped according to stipulated schedules defined by contractual obligations. It was imperative, therefore, to establish accurate and timely controls. The system in general, then, is one of maintaining a record of all inventory status changes, by part number and serial number, from receipt of the product (from assembly work-in-process) to actual shipment. In addition, the control system has the inherent ability to provide immediate replies to such questions as:

> Was the product closed into finished goods in compliance with the manufacturing schedule?
>
> What is the location and condition - test and inspection, rework, shipping inventory, etc. - of the completed products?
>
> Was the product shipped in compliance to contractual schedules?

Input Transactions. The system, as illustrated in Fig. 1, utilizes three Stromberg Time Transacters to record the movement and status of products. One is located in the Finished Goods Stores, the second in the Product Compliance area, while the third is located in the Shipping Office. Three classes of information are entered and recorded simultaneously - a plastic stub card containing the transaction code, a pre-punched part and order identification card, and variable information such as quantities or DD250 numbers which are entered through the transacter dials. By means of this source data recording, faster input and more accurate recording of transactions to the inventory records is thus accomplished.

File Records. The inventory records are maintained on Ramac disk files in two forms, a part number record and a serial number record for end units. The part number inventory records are fairly typical in that they contain summarized information such as the total quantity in rework or in test and inspection. In addition, cumulative totals are maintained for manufacturing and delivery schedules. The part number record contains, as well, part identification codes (end unit, subassembly, or modification kit) and procurement codes.

The serial number record is unique, however, since it contains the latest information on the location and status of each end unit. In addition to providing supporting details for the quantities in the part number record, its primary function is the provision of replies to the types of inventory questions raised earlier. However, these records would be of little value if we could not maintain or interrogate them on a real time basis as provided by random access storage.

Output Reports. In addition to supplying interrogation replies, the control system provides management reports on weekly production and shipping performance - what was manufactured and shipped according to the specific, predetermined plan and what variances occurred. Equally important, a delinquency report by making department is supplied. Other reports are and will be provided, as management requires them, without undue delays. Through the use of report generators, we have the ability of dumping the file contents on tape and process them through the 1401 computer for any conceivable type report utilizing the file data. This can be done more quickly and economically than any technique we have used in the past.

Other Applications

Obviously, Hughes Aircraft Company is not the only company to recognize the advantages offered by new hardware and programming aids. A number of companies, large and small, have implemented or are planning similar systems. Faced with a problem of maintaining and controlling 160,000 tab cards utilized in a spare parts inventory management control application, Convair-Astronautics in San Diego turned to large random-access storage for solution. In addition to local input, the system is geared up to accepting off-site inventory transactions via a five-channel punched paper tape communication system. The result! Cost per transaction posting was reduced by 25 per cent in addition to obtaining a tremendous increase in control and visibility.[5]

Another company, Federal Telephone and Radio Company, having converted to source data recording, reports, ".... over-all (input) costs are much less, reflecting fewer errors, new information, and faster action to clear up shop trouble. Four keypunch clerks have been freed for other assignments. Manual reporting errors and wasted time have been cut to an insignificant level in the plant".[6]

The American Bosch Division of the American Bosch Arma Corporation utilizes large random access file processing for controlling the manufacture and procurement of some 15,000 parts used on approximately 1000 end-products. Processing scope extends from a level-by-level net part requirements computation to the issuance of purchase requisitions and shop fabrication orders. Some of the benefits claimed to have been derived from the system are:

1. Reduction of inspection, ordering and set up costs by cutting the number of manufacturing lots - and related paper - processed through the shop.

2. Increased accessibility to accurate, up-to-date production control records.

3. Indications of inventory shortages in time to take corrective action.

American Bosch states that these and other advantages have already resulted in net savings which are in excess of $120,000 per year.[7]

Conclusion

As indicated by our experience at Hughes as well as others, management information requirements can be met. The hardware and the soft-ware is available. We are remiss in our responsibilities if we do not utilize them to their fullest potential.

References

1. Wall Street Journal, December 27, 1960, "Bugs in Automation"

2. Milton M. Stone,"Data Processing and the Management Information System" (AMA Management Report Number 46)

3. Peter Drucker, "Thinking Ahead" (Howard Business Review, Jan.-Feb. 1959)

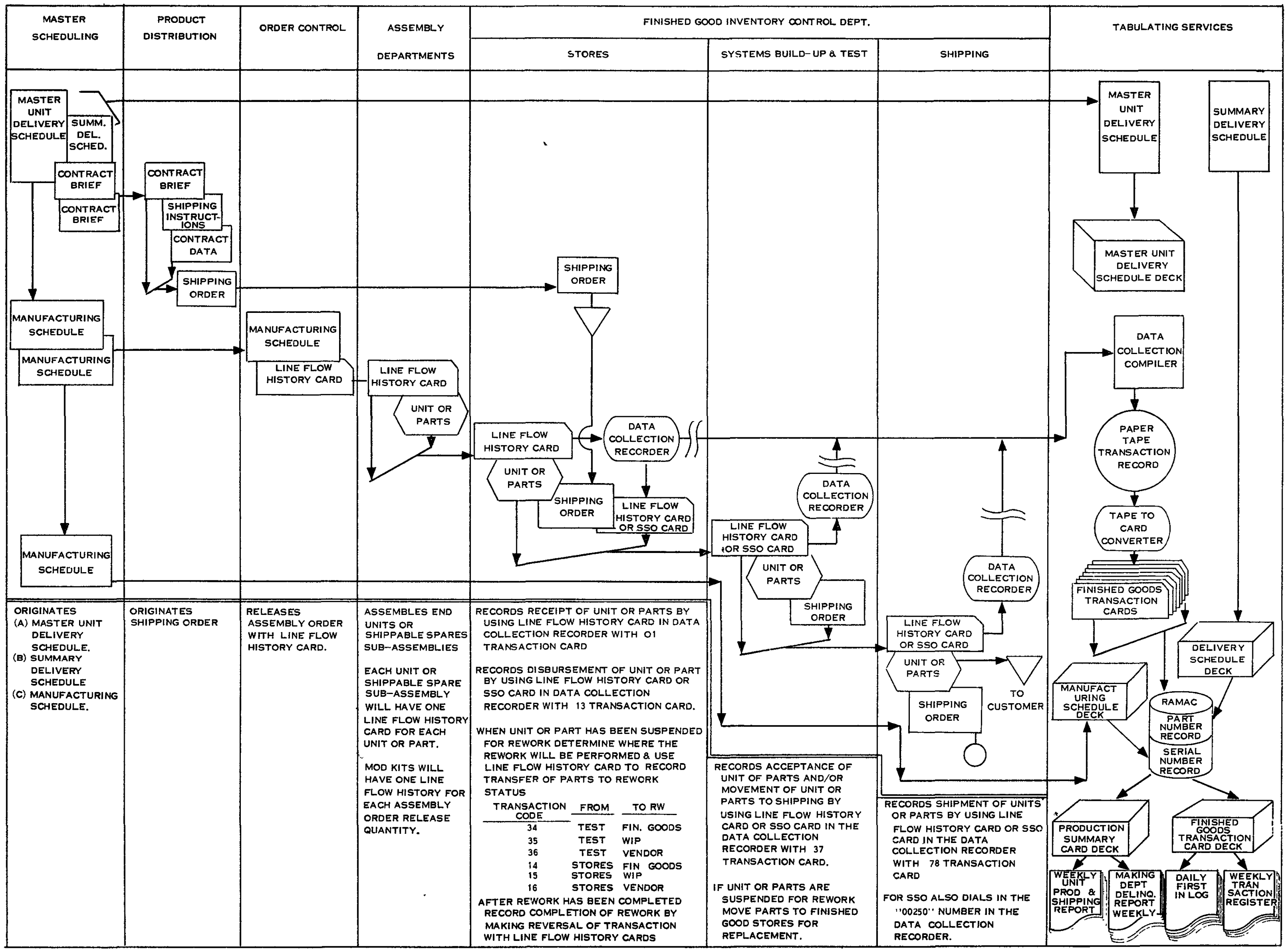
4. Elizabeth Marting, Editor, "Top Management Decision Simulation" (American Management Association 1957)

5. J. A. Dufresne, F. J. Knight, "Spare Parts Control for the Atlas Missile" (Aircraft & Missile Production Management Proceeds, IBM, Feb. 8-10, 1960)

6. R. W. Christian, "Controlled Plant Information" (Factory Magazine, Aug. 1960)

7. "Manufacturing Control at American Bosch Division" (IBM) General Information Manual E20-2053

FINISHED GOODS INVENTORY MAINTENANCE GENERAL FLOW



MASTER SCHEDULING

ORIGINATES
(A) MASTER UNIT DELIVERY SCHEDULE.
(B) SUMMARY DELIVERY SCHEDULE
(C) MANUFACTURING SCHEDULE.

PRODUCT DISTRIBUTION

ORIGINATES SHIPPING ORDER

ORDER CONTROL

RELEASES ASSEMBLY ORDER WITH LINE FLOW HISTORY CARD.

ASSEMBLY DEPARTMENTS

ASSEMBLES END UNITS OR SHIPPABLE SPARES SUB–ASSEMBLIES

EACH UNIT OR SHIPPABLE SPARE SUB–ASSEMBLY WILL HAVE ONE LINE FLOW HISTORY CARD FOR EACH UNIT OR PART.

MOD KITS WILL HAVE ONE LINE FLOW HISTORY FOR EACH ASSEMBLY ORDER RELEASE QUANTITY.

STORES

RECORDS RECEIPT OF UNIT OR PARTS BY USING LINE FLOW HISTORY CARD IN DATA COLLECTION RECORDER WITH 01 TRANSACTION CARD

RECORDS DISBURSEMENT OF UNIT OR PART BY USING LINE FLOW HISTORY CARD OR SSO CARD IN DATA COLLECTION RECORDER WITH 13 TRANSACTION CARD.

WHEN UNIT OR PART HAS BEEN SUSPENDED FOR REWORK DETERMINE WHERE THE REWORK WILL BE PERFORMED & USE LINE FLOW HISTORY CARD TO RECORD TRANSFER OF PARTS TO REWORK STATUS

TRANSACTION CODE	FROM	TO RW
34	TEST	FIN. GOODS
35	TEST	WIP
36	TEST	VENDOR
14	STORES	FIN GOODS
15	STORES	WIP
16	STORES	VENDOR

AFTER REWORK HAS BEEN COMPLETED RECORD COMPLETION OF REWORK BY MAKING REVERSAL OF TRANSACTION WITH LINE FLOW HISTORY CARDS

SYSTEMS BUILD-UP & TEST

RECORDS ACCEPTANCE OF UNIT OF PARTS AND/OR MOVEMENT OF UNIT OR PARTS TO SHIPPING BY USING LINE FLOW HISTORY CARD OR SSO CARD IN THE DATA COLLECTION RECORDER WITH 37 TRANSACTION CARD.

IF UNIT OR PARTS ARE SUSPENDED FOR REWORK MOVE PARTS TO FINISHED GOOD STORES FOR REPLACEMENT.

SHIPPING

RECORDS SHIPMENT OF UNITS OR PARTS BY USING LINE FLOW HISTORY CARD OR SSO CARD IN THE DATA COLLECTION RECORDER WITH 78 TRANSACTION CARD

FOR SSO ALSO DIALS IN THE "00250" NUMBER IN THE DATA COLLECTION RECORDER.

608 - 611 Missing from proceedings

THE COMPUTER SIMULATION OF A COLONIAL SOCIO-ECONOMIC SYSTEM

Warren Dee Howard
Defense Systems Division
General Motors Corporation
Warren, Michigan

The spectrum of weapons employed by the Communists is not confined to force, but brackets all possible relationships between states and social groups---ideological, political, economic, psychological, cultural, technological and military.
Strausz-Hupé

Summary

One might view the international political world as a system of nations, each described by a unique transfer function. Existing system engineering methods and computer techniques might then be applied to this multi-variable system with the hope that a better understanding might be achieved for the rise of international problems and their subsequent solution.

This paper describes the design, and actual demonstration by analog computer techniques, of a colonial socio-economic system which included national growth and national behavior models.

The national growth model included such variables as resource, opportunity and incentive with the hopes of evaluating the asymptotic behavior of the total population. The national behavior model described which one of the two political alternatives would be advocated by elements of the organized native class based on the environment defined by the growth model.

Introduction

In considering the requirements for a "total defense", the modern military strategist must critically evaluate the roles and potentials of factors introduced from the areas of economics, sociology, psychology, etc. This is particularly true today where we, as a nation, are involved in a large number of potentially dangerous international activities known as the Cold War.

If we were to look into our arsenal of these types of weapons, however, we would immediately find that we are quite poorly equipped. There are many reasons for this; but the dominating reason, no doubt, is the fact that the social sciences have not developed to the point where even somewhat valid and reliable predictions can be made. The most often cited difficulty is the reportedly large numbers of variables which must be evaluated and processed.

With the advent of modern computation techniques and machines, however, the processing of large numbers of equations and variables is no longer a problem. Moreover, a new technology generally termed "systems engineering" has developed which has made possible the simulation and solution of large scale systems problems.

One might then speculate on the question as to whether or not the nations of the world might be conceived of as a system of elements which, when quantified, might lead to predictions of future instabilities, growths, etc.

An attempt was made, therefore, to consider the growth and subsequent behavior of a primitive nation in light of the foregoing to determine the feasibility of such a simulation.

Although the social scientists in general have failed to make use of mathematics and hence computers, we cannot ignore the very important work which they have accomplished. The documentation of this work includes a wealth of verbal propositions which attempt to explain some phenomenon in their area of interest. Important then in the construction of a nation simulation, is the possible quantification of these verbal propositions.

By considering small portions of the model on an analog computer, one might intuitively quantify certain of these propositions by comparing the computer output with known data. Each proposition so quantified may then serve as a building block for more complex computer models.

The Model

Development of the Model

The socio-economic model was constructed in several stages on the analog computer; beginning with the primitive group and ending with national behavior. An exhaustive literature search in the social sciences produced the general ideas of how the model might be constructed. From biology,[5,11] came the familiar first order equation which describes the unconstrained development of a species. The thought of a Law of Diminishing Returns was borrowed from early economists[4,8] and the consumption function from more contemporary authors.[6] The national behavior is an application of research done by Rashevsky.[9,10]

For proper selection of initial conditions and constants, many statistical publications were consulted,[2,12,13] and from the documentaries,[1,3] came a wealth of verbalizations which led to the inclusion of certain constraints.

The Model, a Verbalization

The nation is described by two classes of models: (1) national growth, and (2) national behavior. The national growth models define the environment in which a three-class society, consisting of the colonizers, organized and primitive natives, evolve. The national behavior model describes which one of two political alternatives the organized native class advocates, under the influence of the environment as defined by such parameters as resource, opportunity, incentive and technology.

These two classes of models define a nation, the majority of whose inhabitants are considered primitive. These people are seen to exist in a relatively closed society which evolves around a resource, R. The personal needs of the natives are represented by a consumption function which includes a marginal propensity to consume itself depending on the favorability of the times. To add more realism, the primitive work force has its production affected by a Law of Diminishing Returns.

The nation also possesses an incentive which is the motivation for an external power to emigrate personnel to the nation for the purpose of exploitation. The colonizers organize a portion of the primitive population to serve as the labor force for this exploitation. The organization of the natives is seen as mutually beneficial, and an opportunity factor is thereby created for the primitives; hence, a new organized class is formed.

The organized natives are visualized as those who will form the urban areas, become more highly educated, and in general engage in those activities which by our standards would tend to improve their standard of living from what it was in the primitive society.

The work force represented by a portion of the organized natives develops the incentive for the colonizer. The effectiveness of their work is controlled by a worker efficiency function which takes into account an optimum management labor ratio.

Figure (1) shows the organizational structure of the three classes. The primitives are seen to be organized along the traditional tribal lines. The colonizers are depicted as being of the leader type; because of the nature of their work, administrative. The organized natives develop into a large follower class whose allegiance is sought by two opposing leader groups; one which advocates status quo, the other a change.

By constraining the technological level of the nation, the incentive is, with time, depleted. With the disappearance of incentive, the opportunity for new individuals entering the organized society is diminished. With the loss of opportunity, the size of the leader group which advocates a change will grow; hence, the total number of people in the organized class advocating a change will grow. The rate and ease with which the behavior of the organized group changes is developed as a function of the relative number of leaders, followers and their respective "influence coefficients". It is suggested that the latter themselves are functions of communication, transportation; hence the technological level of the nation.

Thus, we have described a model which permits the study of a colonial socio-economic system, not with the desire of fitting the present model to a specific nation, but to demonstrate that you can take apparently pertinent variables and construct a dynamic model whose characteristics are somewhat realistic.

The Primitive Society

Let us now consider how a simple model was constructed. First we have a group of primitives, and we should like to describe how they evolve. From various levels of organisms we find that their development may be described by the following system of equations:

$$\frac{d\eta_i}{dt} = F_i(\eta_1, \eta_2, \cdots \eta_n) \qquad (1)$$

where the η_i are elements of the society itself. It is sometimes possible, however, to consider a system of a single variable η. The above system of equations then reduce to

$$\frac{d\eta}{dt} = F^*(\eta) \qquad (2)$$

The previous assumption holds when for any reason one species or group grows actively while conditions otherwise remain substantially constant. This seems to be essentially what has occurred in most human populations. It is true that in their growth, they have carried along with them a complicated industrial system or group comprising both living and non-living elements. We shall, however, look upon the number of the human population itself as a sort of single index or measure of the group as a whole.

To be more correct, equation (2) may take the following form:

$$\frac{d\eta}{dt} = (b-d)\eta \qquad (3)$$

where

b is per capita birth rate

d is per capita death rate

This form, however, was no more amenable to modeling than equation (2) so it was decided that equation (2) slightly modified would be the core of the primitive society.

$$\frac{d\eta}{dt} = \eta[F(\eta)] \qquad (4)$$

As the primitives were viewed as a group with neglible external influence and a relative low and constrained technological capability, in equation (4) was seen as primarily a function of what the natives considered as necessary resources. We shall close the loop in this class as shown in figure (2). The primitives will develop and use a fixed resource which exists within the sphere of their influence according to laws which we shall now describe.

Consumption Function

As the ultimate desire is to construct a model which might describe an African nation, the habits of the more primitive groups here were noted. Cloete[1] points out that the typical African will generally produce only to his needs, and that he does not tend to stockpile. With this in mind, the consumption function of the primitives was set as a per capita constant plus a marginal amount which depends upon a favorable difference between production and consumption.

The constant was viewed as the minimal requirements for the group under unfavorable conditions; namely when $\frac{d\eta}{dt}$ was negative.

To simplify the fitting of the model on existing analog computing equipment, the marginal propensity to consume was set as follows:

$$\text{for } \begin{array}{ll} p<c & \Delta c = 0 \\ p>c & \Delta c = k(p-c) \end{array} \qquad (5)$$

Production Function

This portion of the model presents a little more complication. First there must be a "production force" whose numbers may be different from that of the total population. Secondly, we must consider the Malthusian[4] concept of the Law of Diminishing Returns as applied to resource.

Again, to facilitate the fitting of the present model on existing computing equipment a simple approximation was used:

$$\eta_{(P)} = \eta_{(t-\tau)} \qquad \text{for } \frac{d\eta}{dt} > 0$$
$$\eta_{(P)} = \eta_t \qquad \frac{d\eta}{dt} < 0 \qquad (6)$$

where

$\eta_{(P)}$ is production force

η_t is present population

$\eta_{(t-\tau)}$ is the population τ years previous

It should be pointed out that a more sophisticated approach to the above approximation would be to consider the age distribution of the primitives. Sharpe[7] has shown that there is a certain stable type of age distribution about which the actual age distribution varies, and towards which it returns if a perturbation (nature or man) causes any deviations. As the degree of sophistication of our model heightens, the necessary increase in computing components will be made to include this important work.

Now let us consider the production which is accomplished by the "production force". This output may be noted as follows:

$$P = k\,\gamma\,\eta_{(P)} \qquad (7)$$

where

P is production of resource per unit time

$\eta_{(P)}$ is production force

k is the amount of labor per unit time that is willing to give

γ Law of Diminishing Returns

In general, all the terms of equation (7) are amenable to measurement except γ. It can be seen that γ in general will be a function of both $\eta_{(P)}$ and resource.

$$\gamma = \gamma(\eta_{(P)}, R) \qquad (8)$$

As an approximation, we shall assume that there exists a critical ratio $\eta^*_{(P)}/R$ which determines the maximum efficiency per unit resource subject to the following:

$$\begin{array}{ll} \eta_{(P)} < \eta^*_{(P)} & \gamma \text{ is increasing} \\ \eta_{(P)} > \eta^*_{(P)} & \gamma \text{ is decreasing} \\ \eta_{(P)} = 0 & \gamma = 0 \\ \eta_{(P)} = \infty & \gamma = 0 \end{array} \qquad (9)$$

Thus γ was chosen as an arbitrary function of the general shape of a Poisson distribution and was equipped with the capability of varying $\eta^*_{(P)}/R$.

Resource-Saving

As can be seen from figure (2) the consumption and production rates are summed, and their effects are applied to the unused resource. By the very nature of the way in which we conduct our lives (periodically sleeping, working, etc.), the actual process of summation of rates is viewed as a sampled process; however, for our purposes we chose to allow the time integral to approximate this process.

$$\sum_{i=0}^{t} (p_i - c_i) \cong \int_0^t (p-c)\,dt \qquad (10)$$

The constant k_2 which appears in figure (2) is itself a function of the technological level of the group.

Constraining $F(\eta)$

It seemed reasonable to assume that the initial magnitude of the resource R should affect $F(\eta)$ only within certain limits for two reasons.

1. The magnitude of R subtly assumes nature's effect on R over the period of evolution, and hence R's magnitude may not be known to the primitives at any given time. They probably see either favorable or unfavorable conditions.
2. The constraint on their technological level and their assumed group characteristics leave them without the desire or the tools to measure R even if they so wished.

$F(\eta)$, more properly, should be constrained in the favorable condition; however, it should be left unconstrained for the unfavorable condition. To see this, we must first justify allowing R to go negative. Previously we referred to R as being considered necessary resources. It is felt that when the magnitude of R knowingly to the primitives approaches zero (possibly below the value of positive constraint), that the primitives will modify their total consumption of resource to include items which were not previously considered as resource. This would imply a certain ingenuity on the part of the primitives and would also require a learning period. As the availability of this Δ resource would be constrained by nature, the unfavorable condition imposed by R going negative would be conducive to emigration and famine; hence $F(n)$ may take on large negative values for short periods.

Development of the Organized Society

The model which describes the organized society is similar to the one which describes the primitive society with two exceptions. First the Law of Diminishing Returns is replaced with a production efficiency function. Second, the resource element is replaced by an opportunity element.

The modification of the γ function is quite straight forward and essentially requires the substitution of $n^*_{(P)}/n_c$ for $n^*_{(P)}/R$ with the conditions already enumerated in equation (9) still holding and where n_c is the number of colonials.

The second change is more one of definition than anything else. There is, however, no attempt to constrain the amount of opportunity remaining for the group.

Development of Colonizing Society

The presence of an undetermined quantity of incentive is the motivating factor for the immigration of a group of colonizers who are capable of organizing a portion of the native population. An equation of the form (3) is again used with $F(n)$ being a constrained function of the remaining incentive. Whether or not the incentive is allowed to go negative might well be considered as government policy towards the nation.

The rate of change of incentive is considered to be of two factors: (1) a constant demand, and (2) a productive output.

$$\frac{dI}{dt} = k_3 + c\, n_{(P)}\, f\left(\frac{n_{(P)}}{n_c}\right) \qquad (11)$$

where

k_3	is constant demand
$n_{(P)}$	is the number of organized working natives
$f\left(\frac{n_{(P)}}{n_c}\right)$	is an organizational efficiency function
c	is a work constant

The constant demand is viewed as that drain on the incentive which would be caused by governmental policy, both within and outside the colony, changes in value systems, etc.

The second term of (11) is quite straight forward with $f(n_{(P)}/n_c)$ being arbitrarily chosen of the same form as γ in equation (8).

Thus far we have described a model of essentially a three-class society. It is reasonable to assume that there will be negligible interclass mobility between the colonizer and the other two groups; and as a close approximation, it may be neglected between the organized and primitive groups if, again, the end result is to correspond to an African society. This may be seen by noting that mobility is a function of transportation among other items, and it is noted that public conveyances in the areas of consideration that would be used by the individuals passing from one class to another tend to have the same load factors whether they are leaving or returning to the urban area. If there was a differential mobility between these two classes, it would show up in the load factors; and if the mobilities are equal, then they can be safely neglected.

National Behavior

As we previously indicated, our center of attention is the organized group as to which one of two mutually exclusive behaviors they exhibit; status quo or change. The fundamental concept used here is a generalization on Rashevsky's theory of imitative behavior of a social group. Interested readers are referred to the original publications[9,10] for the development of this theory.

In general the approximation takes the following form:

$$\frac{dn_5}{dt} = a_1 n_1 + a_2 n_2 - a_3 n_3 - a_4 n_4 \qquad (12)$$

where

n_5	total of those advocating status quo
n_1	active (leader) individuals advocating status quo
n_2	passive (follower) individuals advocating status quo
n_3	passive (follower) individuals advocating change
n_4	active (leader) individuals advocating change

a_i influence coefficients

The influence coefficients are primarily functions of communication (number of lines of newsprint, hours of radio broadcasting, etc.), and it may be noted that in general the influence coefficients associated with the active individuals will be of greater magnitude than those associated with the passive types; for the actives will, in most cases, be in control of the press and radio.

A similar expression exists for $\frac{d\eta_R}{dt}$:

$$\frac{d\eta_R}{dt} = a_3\eta_3 + a_4\eta_4 - a_1\eta_1 - a_2\eta_2 \qquad (13)$$

where

η_R is the total of those advocating a change

It now becomes necessary to modify slightly the nomenclature used in identifying the individuals in each class, for we have to consider an interclass effect.

η_c	number of colonials
η_I	number of primitives
η_{II}	number of organized natives
η_{II}^c	number of organized natives advocating actively status quo
η_{II}^I	number of organized natives advocating actively change

Equations (12) and (13) can be reduced to a single equation in η_s :

$$\frac{d\eta_s}{dt} = (a_1 + a_3)\eta_2 - \{a_3\eta' - a_1\eta_1 + a_4\eta_4\} \quad (14)$$

where
$$\eta' = \eta_{II} - \{\eta_{II}^c + \eta_{II}^I\}$$
$$\eta_1 = \{\eta_c + \eta_{II}^c\}$$
$$\eta_4 = \eta_{II}^I$$

The intraclass group structure is defined as follows:

$$\eta_{II}^c = \gamma_{II}^c \eta_{II}$$
$$\eta_{II}^I = \eta_{II}\{1 - C(I)\} \qquad (15)$$

The constant γ_{II}^c determines the initial percentage of organized natives who are actively advocating status quo. The constant is amenable to measure and is of the order of 3%. $C(I)$ is a constrained function of opportunity which allows the number of individuals actively advocating a change to increase as opportunity approaches zero.

As Rashevsky has extended his theory to include n mutually exclusive behaviors, one might speculate on the possibility of construction of a behavior model of the Congo with three mutually exclusive behaviors, say M, K and L, such that the effects and potential stability of a coalition between M and K might be studied.

Discussion

Figure (3) shows a sample run of the socio-economic system. This experiment was run to determine the feasibility of applying a systems engineering approach to the simulation of a nation. The general dynamic characteristics of the model leave one highly optimistic concerning this hypothesis. It is interesting to note the development of the native society is comparable with many present day colonial systems. The constraints placed on the technological development appear to be substantiated as far as the remote primitives are concerned; but, by the very definition of the organized natives, it would appear that the constraint is not realistic.

Many of the constants which appear in the model are inherently statistical and are not amenable to quantification, such as the influence coefficients in the national behavior model which are, by definition, mean-values of probability distributions.

To strengthen the present model, the following modifications will be incorporated:

1. The application of Monte-Carlo techniques to those portions of the model which are inherently statistical, such as equation (14).
2. Removal of the constraint on the technological level, especially in the colonizer and organized native groups.
3. The replacement of specific group structures with appropriate distribution functions.

Conclusion

Through the use of building block techniques on an analog computer, a model of a colonial socio-economic society was developed. In general, the colony is described by two classes of models: (1) national growth and (2) national behavior. The growth models lead to a three-class society, each of which develops as a function of their numbers and respective value system.

The national behavior model describes which one of two political alternatives is advocated by the organized native class. The rate of change of group size advocating a particular behavior is shown as a function of the relative number of leaders, followers and their respective "influence coefficients". It is suggested that the latter are themselves functions of communication, transportation; hence the technological level of the nation.

Two important points have been demonstrated:

1. The potential of the analog computer in the possible solution of the problem.
2. The possibility of modeling the evolution of society through building block techniques.

It would be interesting to speculate on whether or not this approach might be the key to the quantification and subsequent unification of the social sciences.

References

1. Cloete, S., The African Giant, Boston: Houghton Mifflin, 1955.
2. Department of Economic & Social Affairs, The Future Growth of World Population, New York: United Nations, 1958.
3. Gunther, J., Inside Africa, New York: Harper, 1955.
4. Heilbroner, R., The Worldly Philosophers, New York: Simon & Schuster, 1953.
5. Lotka, A., Elements of Mathematical Biology, New York: Dover, 1956.
6. Murad, A., Economics, Ames, Iowa; Littlefield, Adams, 1958.
7. Sharpe, F., "Normal Age Distribution", Philosophical Magazine, April, 1911 p 435.
8. Smith, A., The Wealth of Nations, New York: Random House, 1937.
9. Rashevsky, N., Mathematical Theory of Human Relations, Bloomington, Indiana: Principia Press, 1947.
10. Rashevsky, N., Mathematical Biology of Social Behavior, (rev ed) Chicago: University of Chicago Press, 1959.
11. Rashevsky, N., Mathematical Biophysics, (3rd ed) New York: Dover, 1960.
12. Statistical Office of the United Nations, Handbook of Population Census Methods, 3 vols, New York: United Nations, 1958.
13. Statistical Office of the United Nations, Statistical Yearbook 1959, New York: United Nations, 1959.

SOCIO - ECONOMIC MODEL
INTRA-CLASS GROUP STRUCTURE

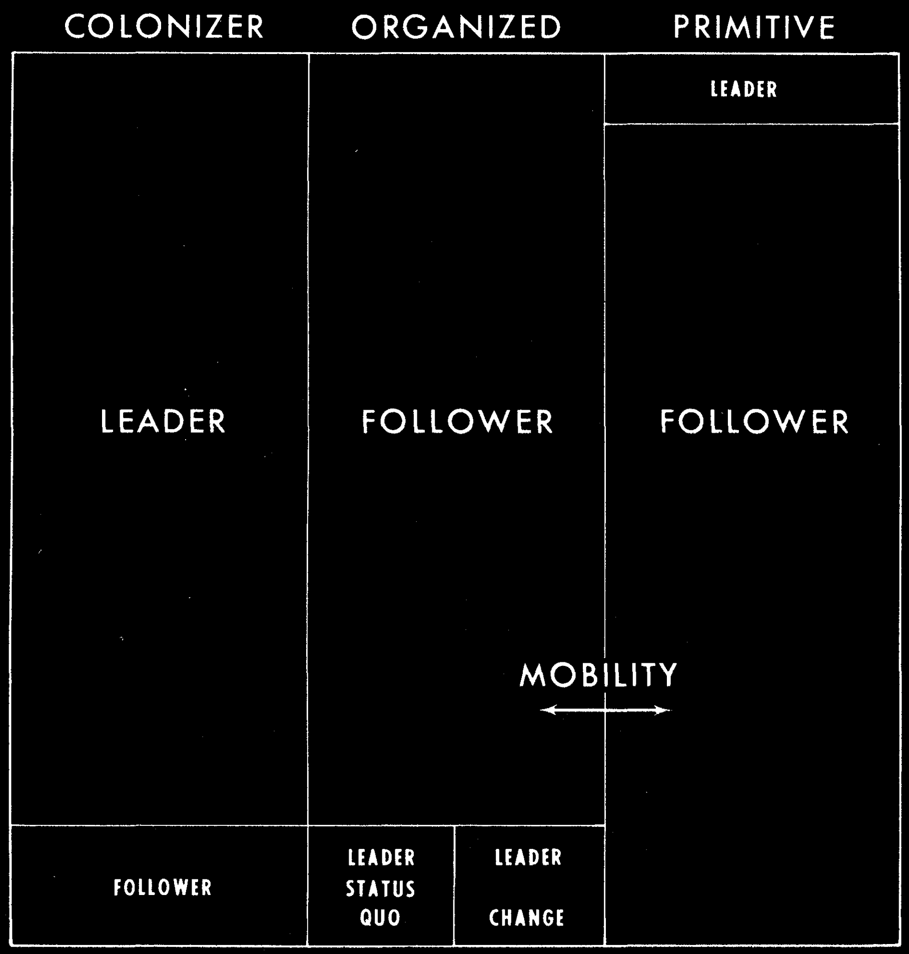
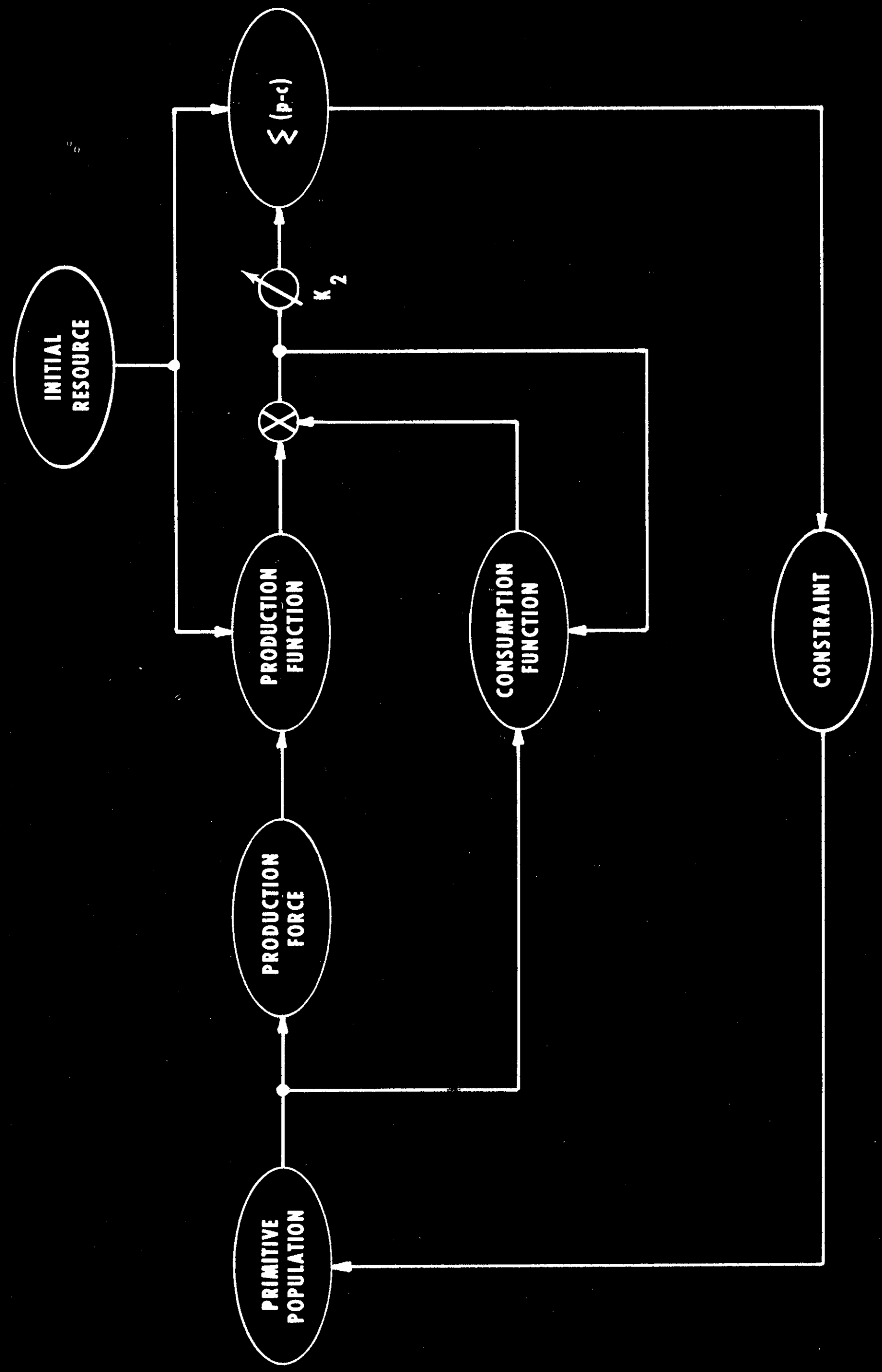


Figure 1



INTRA-CLASS GROWTH MODEL

Figure 2

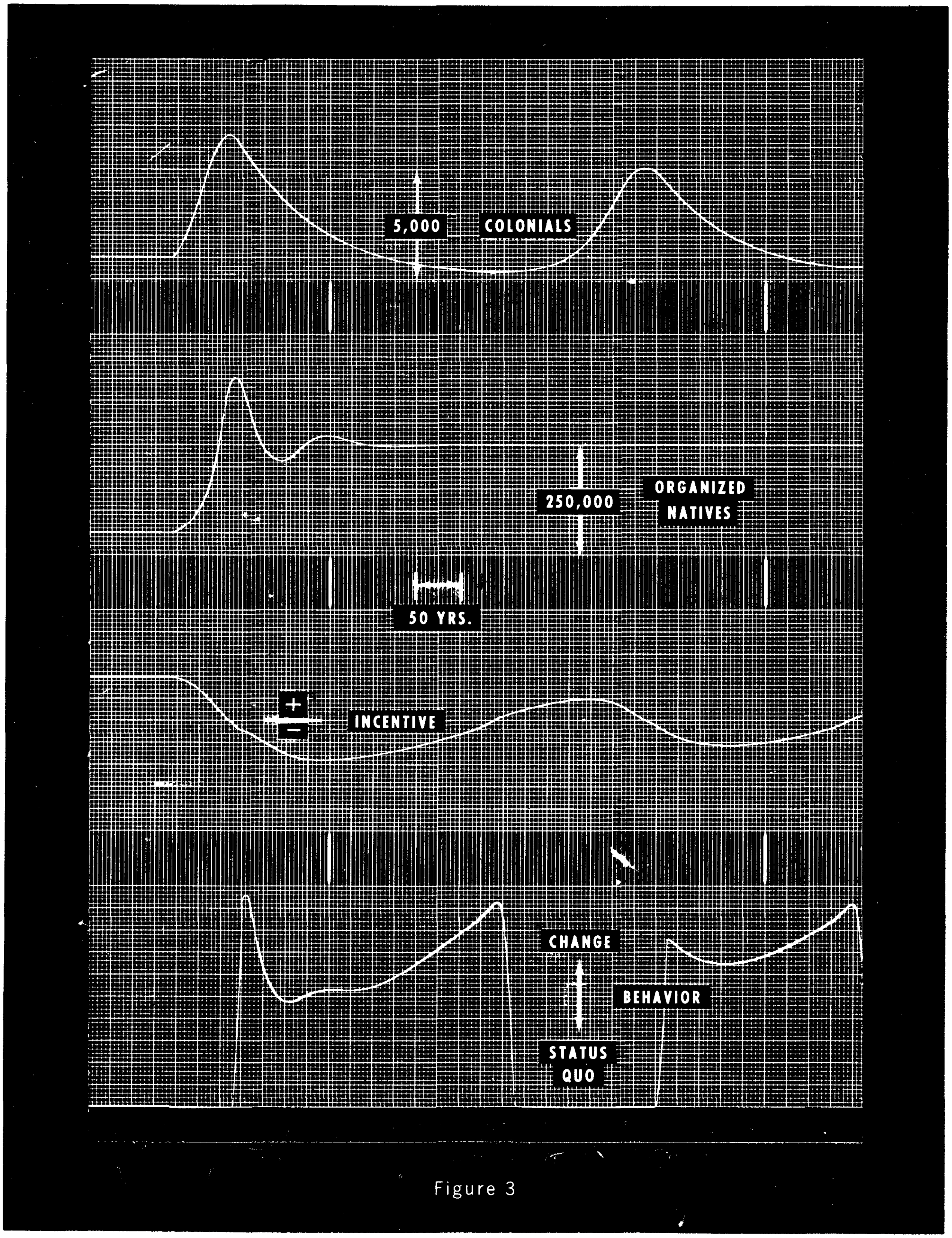

5,000
COLONIALS
250,000
ORGANIZED NATIVES
50 YRS.
+
−
INCENTIVE
CHANGE
BEHAVIOR
STATUS QUO

Figure 3

X-15 ANALOG FLIGHT SIMULATION PROGRAM - SYSTEMS DEVELOPMENT AND PILOT TRAINING -

N. R. Cooper
North American Aviation, Inc
Los Angeles 45, California

The extensive requirements for simulation in the design of the X-15 Research Vehicle became apparent very early in the development program. In early 1956, only weeks after NAA had been awarded the X-15 contract, a simple analog mechanization was being utilized, together with a crude three-axis controller to define the requirements for a manual reaction control system, a problem area which at that time was as new as space rendezvous is today. However, our most exaggerated estimates at that time regarding the application of simulation techniques in the development and flight testing of the X-15 are seen now in retrospect to have been most conservative. It may be said that the X-15 simulation requirements literally "grew" with the program. The flight simulation has been used in all design and testing phases; first to support the configuration and subsystem design; next to test and evaluate subsystems, displays, and control hardware, both in prototype and in final production form; and finally to provide flight test support and pilot training. This paper will review briefly the overall simulation program conducted during the 5-year development of the X-15 and will describe in detail the unrestricted six-degree-of-freedom mechanization which has been utilized during the past 3 years of the program. It will describe the most important of the many applications in which the simulator has been, and in fact is still being, utilized in the design, development, and flight testing of the X-15. Finally, it will be interesting to compare the simulation requirements of the X-15 with those of more advanced space vehicles and attempt to interpret problems or deficiencies encountered in the X-15 simulation as they may affect more advanced space mission simulation.

A review of the X-15 vehicle and its intended mission give an insight into the simulation requirements associated with the program. Figure 1 shows the X-15 vehicle and the control system configuration. The X-15 has an upper and lower movable vertical stabilizer for yaw control, and differentially operated horizontal surfaces for pitch and roll control while in the atmosphere. The reaction system for attitude control outside the atmosphere consists of small hydrogen peroxide rocket motors in the nose and wingtips. Electronic stability augmentation in the X-15 consists of rate damping about all three axes.

Figure 2 reviews the design mission and performance capability of the X-15. It is capable of attaining speeds in excess of 6000 feet per second, or Mach 6.0, and altitudes to 500,000 feet. Very early in the X-15 development it became apparent that a six-degree-of-freedom simulation, covering as much of the complete flight regime as possible, was necessary in order to explore the problems of manually controlled boost, re-entry, and space flight. In past airplane designs, stability and control problems could be adequately analyzed and solved by utilization of a five-degree-of-freedom (constant-velocity) simulation at a large number of given flight conditions. The addition of the sixth degree of freedom, namely the variation in velocity and altitude, entailed much increased complexity in the mechanization required and was not warranted, since the effects of these variations were not significant in the overall stability analysis. However, the X-15 can transverse its flight profile at a rate much greater than any encountered before. The rates of change of velocity and altitude, and correspondingly of dynamic pressure, during the boost and re-entry phase of the X-15 mission become large enough to greatly influence stability and control characteristics. The performance range of the X-15 makes the number of flight conditions which would be required to be investigated on a five-degree-of-freedom simulation prohibitively large. Also, it was necessary to allow the pilot to evaluate all the various phases of the mission during continuous simulated missions to establish proper compatibility with the human of the control systems, instruments, etc, during design. Thus, the requirements for at least a limited six-degree-of-freedom simulation was obvious purely from a stability, control, and human engineering standpoint, without regard for the many added advantages it would provide in pilot training, mission evaluation, and flight planning.

Because of the need for adequate aerodynamic data and early limitations of analog equipment, it was not expedient to initiate a six-degree-of-freedom simulation immediately. Consequently, during the first year of the program, efforts were directed toward specialized problems which could be solved on simpler simulations. Figure 3 presents a time schedule of the various simulation activities from the initiation of the X-15 program to the present time. It is of interest to note the increasing sophistication of these studies. It is also interesting to note that the problems of early interest were those about which we knew the least, namely the exit and re-entry problem and the problems associated with reaction control. As mentioned previously, the simple reaction control mechanization was initiated in early 1956 and provided preliminary criteria for the reaction control system design. This was followed by a three-degree-of-freedom longitudinal mode mechanization of the exit and re-entry flight characteristics, utilizing very early aerodynamic data, for preliminary evaluation of control characteristics during these phases of the mission. This program showed that, in the longitudinal mode at least, manual control without augmentation was possible, but that augmentation was desirable. Next a "programmed q" analog mechanization was utilized to evaluate the compatibility of reaction and aerodynamic controls during the transition phases of exit and re-entry. Reaction control duty cycle requirements were also finalized. This simulation programmed the dynamic pressure variations of a design altitude mission and utilized constant aerodynamic data estimated at Mach 6. During the early simulation studies, the ability to include the pilot in the loop was provided by means of a simple cockpit simulator which included a reaction control stick.

The first six-degree-of-freedom mechanization which was utilized in the X-15 program was limited to Mach numbers greater than 2 and altitudes greater than 50,000 feet in order to simplify the mechanization. In this manner we were able to eliminate the complex variations in aerodynamic characteristics in the transonic regions and to minimize the range of variations of air density which had to be simulated. This simulation was utilized for approximately 8 months during 1957. During this period, the major design and development of the primary control system and the stability augmentation system on the X-15 was accomplished. Also, additional stability requirements were determined, and several design changes were incorporated in the basic configuration. The five-degree-of-freedom analog program shown on the schedule was run simultaneously with the early six-degree studies in order to evaluate stability and control characteristics, including roll coupling, at subsonic and transonic conditions.

One other simulation program which was accomplished during this period is worthy of note before we discuss the complete six-degree-of-freedon simulation. Since there was some concern as to the pilot's ability to control the X-15 under the dynamic loading conditions characteristic of exit and re-entry, a simulation program was conducted in the Human Centrifuge at the U.S. Naval Air Development Center, Johnsville, Pennsylvania, during the summer of 1958. In this program, the centrifuge was driven by computed signals from a limited six-degree-of-freedom analog simulation of the X-15 which was very similar to the one just described. Figure 4 shows, schematically, the method in which the analog computer was used to drive the centrifuge with a pilot operating as a closed-loop controlling function. The centrifuge was driven to follow the computed values of accelerations resulting from pilot inputs and airplane stability and response characteristics, so that the pilot was subjected to the actual G-loads of a specific mission or maneuver. During the centrifuge program, a total of 287 dynamic flights, consisting of the boost and/or the re-entry phase, were accomplished by the seven participating pilots. The results of this program showed that the pilot could control the X-15 during its most severe re-entry conditions, and further showed that, for the X-15 missions, static simulator results were not significantly different from those obtained under dynamic conditions. Consequently, no additional dynamic simulation effort has been necessary in the X-15 program.

Although the limited six-degree-of-fredom simulation provided invaluable information in early design of the airplane, it was soon apparent that it was inadequate. Because of limitations imposed on Mach number and altitude, problems associated with launch and roundout, space positioning, subsonic and transonic stability, and landing could not be investigated. By this time the potential value of the simulator in flight test mission planning and pilot training was apparent, and in these applications, complete simulation capability over the X-15 flight regime would be required. Therefore, the unlimited six-degree-of-freedom simulation was developed and has been in operation continuously since March of 1958. This simulation is capable of piloted or nonpiloted analysis throughout the X-15 flight regime, including launch maneuvers, boost and exit phase, reaction controlled ballistic flight, re-entry, and glide to landing flareout. The performance capability of the simulation covers the Mach range from 0.2 to 10.0 altitude to 500,000 feet, and angles of attack to 35 degrees. Simulated flights may be accomplished with speed brakes open or closed, with flap and landing gear extension effects, ventral jettisoning characteristics, and other nonlinear effects. Missions may be flown using either the XLR-11 or the XLR-99 engine, including the effects of engine throttling and of thrust misalignment. Missions may be accomplished from the design B-52 carrier aircraft or from other advanced carriers such as the B-70. The dynamic effects of nonlinear changes in mass and moments of inertia during engine burning are simulated. Premature burnout of either engine may be simulated at any time, and propellent jettisoning may be accomplished at the required rates.

This simulation capability permits evaluation of all important contributions to the complete mission, with the exception of temperature. A simplified computation of skin temperature at critical points on the vehicle was mechanized on the analog computer and incorporated in the real-time problem solution early in the program. Temperature at one of several points on the vehicle was displayed to the pilot as an aid in remaining within limits during re-entry. However, it was found that this information merely told the pilot that he had exceeded limits and did not provide sufficient lead time to allow corrections once it was established that an over-temperature condition was imminent. Due to the limited application for which the temperature mechanization was suitable, and the equipment required, this part of the simulation was discontinued.

The capability of permitting piloted flights of the X-15 with this simulation is obtained by integration of the mechanization into the X-15 flight control simulator. This simulator is shown in figure 5. It consists of an exact duplication of the airplane cockpit, instruments, and control system. The complete operational flight control system provides exact system characteristics under operating conditions. Actual production components, including cables, push rods, bell cranks, the hydraulic system, artificial feel, etc, are utilized in exactly the same manner as the actual airplane. The electronics of the X-15 stability augmentation system are also included. The control system is duplicated in this detail in order to include all nonlinearities in closed-loop systems evaluation, and to provide the pilot with the exact feel characteristics of the airplane. All control displacements are available to the analog computer by means of electrical pickoffs on each aerodynamic control surface. The cockpit area, shown in figure 6, is a realistic simulation of the airplane configuration. The simulator has the same provisions for aerodynamic control as in the airplane, utilizing either the center stick or the right-hand console stick and rudder pedals. Reaction control is provided with the left-hand three-axis controller. The pilot has normal control over the stability augmentation system by means of the same controller panel as is installed in the air vehicle. The flight instruments, which are simulated in complete operational form, are driven by voltages from the analog computer. They include the inertial attitude indicator which provides pitch, roll, and heading; the inertial velocity, altitude, and rate-of-climb instruments; angles of attack and sideslip; roll rate; normal load factor; indicated airspeed; and for simulator test purposes only, an indication of dynamic pressure.

The equations utilized to simulate the X-15 in the unlimited six degrees of freedom are presented in figure 7. These are basically the classical equations of motion of the aircraft with respect to an Eulerian frame of reference. The equations and all aerodynamic parameters are mechanized in a body-axis system of coordinates. Orientation of the gravity vector and the geographical distance and position are obtained by means of conventional Eulerian angles. Due to the limited ground distance traveled by the X-15 in its design mission, a flat earth may be assumed as a reference. However, the centrifugal acceleration effect is included as a function of the X-15 horizontal velocity component and added directly to the normal gravity vector which is mechanized as a function of altitude. The terms and the equations which are considered variable include thrust, mass, velocity, gravity, inertia, and all aerodynamic coefficients which may not be assumed constant within the accuracy desired.

The analog computer complex used in the X-15 simulation is shown in figure 8. The linear equipment consists of five Model 16-31 Electronic Associates analog computers, incorporating 330 operational amplifiers. Additional nonlinear equipment required in the mechanization includes approximately 80 diode function generators, 25 computing servos, and three electronic multipliers; part of this equipment is shown in figure 9. The nonlinear variations of stability derivatives with Mach numbers and angles of attack are accomplished on four special interpolating servos which provide 17 interpolating points selected, as necessary, over the Mach range for each of the derivatives. The nonlinear variation of the derivatives with the angle of attack at the required Mach number points are obtained from a rack of 60 fixed-base diode function generators.

The greatest utilization of the simulator is obtained when the pilot is included as part of the control loop. Before discussing specific applications of the simulator in the X-15 program it might be well to review a complete simulated flight of the X-15. Typical analog traces of a piloted design altitude mission as flown on the simulator are shown in figure 10. The flight begins at drop conditions, at Mach .8, at approximately 45,000 feet. The throttle is opened immediately after drop, and the pilot makes a pullup to $\alpha = 8$ degrees, which is maintained until the proper climb angle is established. (For a 250,000-foot-altitude mission, this angle is 30 degrees.) Pitch angle is then held constant until burnout, at which time an $\alpha = 0$ ballistic trajectory is established. It is well to note that this represents only one of several acceptable techniques for performing the exit phase. During the period of engine burning, control is required to correct for thrust misalignment. Burnout occurs approximately 90 seconds from drop, at a velocity of 6200 feet per second, at 160,000 feet. The effects of thrust misalignment are seen in the oscillations in angle of attack and sideslip. At burnout, the pilot begins use of the reaction control and continues this means of control throughout the ballistic phase of the trajectory. The recovery used in this particular flight was an angle of attack of 15 degrees, established at approximately 200,000 feet during re-entry. The required aerodynamic trim for this attitude can be set at any time, and the reaction control system then used to establish the required angle of attack. As the dynamic pressure builds up, the load factor is seen to increase, and for this mission, the pilot performed the recovery with the maximum of 5 G. Following the successful recovery, which occurs in this case at approximately 85,000 feet at approximately Mach 5, the altitude is held constant as the airplane decelerates until the desired descent speed is reached.

Consider now what this complete flight simulation allows in the way of system development and testing. Since the pilot can essentially fly the complete mission, a complete evaluation of controls, displays, and augmentation devices is possible long before flight. In the case of the X-15, the complete flight control system, including the stability augmentation system, was operating in the simulator a year before the first flight. The stability augmentation system, which was originally designed and specified on the limited 6-degree simulation was later included in a closed-loop simulation in its exact prototype form on the complete mechanization. These simulation tests on the complete system revealed several inadequacies due to the nonlinear characteristic of the control system; these were corrected before flight. Other changes which were made in the X-15 as a result of simulator testing included control system feel modifications, display changes, and a redesign of the right-hand console grip.

The versatility of the combined simulator and analog mechanization has been more recently demonstrated by the design evaluation and hardware testing of two separate autopilot systems on the simulator. These were both complex adaptive-type autopilots which included attitude-hold modes, reaction and aerodynamic control integration, and other refinements to the basic X-15 control system. These autopilots were included as a part of the closed-loop control system, and their performance was completely evaluated under conditions of the X-15 mission profile. Considerable design changes and improvements were made as a result of these tests, and at the present time, one of these advanced systems is being readied for X-15 flight evaluation. During flight testing of this system, a simulator support program will provide the pilots with preflight training and familiarization with its operational characteristics.

In addition to systems development and stability analysis applications, the simulator has proved invaluable in studies involving the pilot, his control requirements and training, and in flight test planning and support. An example of these is the early development of reaction control techniques. Since the basic X-15 originally had no provisions for stability augmentation through the reaction control system, the control characteristics were those of a perfectly neutral, undamped system unlike any vehicle with which pilots were familiar. Prior to establishment of final reaction control fuel requirements, several pilots were trained on the simulator to perform this type control, and a realistic duty cycle was then obtained. It is significant that gross reductions in fuel requirements were obtained after a pilot had experience on the simulator. Also, it was found that only a minimum amount of experience was required (10 to 20 "flights") for the pilot to reach his near-optimum degree of efficiency.

More recently, the simulator has been utilized by the Air Force, NASA, and North American almost exclusively for pilot training and flight planning. In order to fully appreciate the value of a simulator in this capacity, it is necessary to consider in some detail the X-15 flight test program. First, of course, the cost of an X-15 flight is quite high, requiring that all possible benefit be derived from each mission in terms of research data. The cost factor is also reflected in the necessity for taking reasonably large steps in performance during the flight buildups, which requires that each flight be thoroughly evaluated prior to its acceptance. The safety aspect of research flight planning provides the greatest single requirement for adequate simulation. The simulator allows the research pilot to thoroughly evaluate all aspects of a proposed flight, including all conceivable malfunctions or emergency situations.

The ground positioning, or navigational problem, is a critical consideration in flight planning due to the limited glide range of the X-15. This problem is analyzed on the simulator, with the aid of a detailed map of the flight area (figure 11) on an X-Y plotter upon which the simulated flight path is plotted. This permits the pilot to actually fly simulated emergencies, such as premature engine shutdown, at all points along the proposed flight path, and determine the optimum approaches to the selected emergency landing sites. It is interesting to note that the plotting board used in the simulation is identical to the ones used at the ground station during an actual flight, except of course that, at the actual control station, the flight path is plotted from radar tracking signals rather than from analog signals. This permits the flight test engineer to review, or rehearse, the proposed flight with the pilot on the simulator.

As a result of 3 years of continuous operation of the X-15 simulator in many various applications, certain general conclusions may be made regarding operation, accuracy, and efficiency of complex analog simulations. The six-degree-of-freedom analog simulation has been operated in excess of 10,000 hours in support of the X-15. Over 5000 hours have been logged in the flight control simulator. The simulator is generally on a two-shifts-per-day schedule, and on this basis, an operational utilization of approximately 85 percent is realized. Down time may be attributed equally to equipment problems and to checkout. It has been found that, for a computer complex as large as this, daily problem checks are a requirement. So-called "dynamic" checks, or stability checks, are made by comparing the simulator with IBM-computed transients obtained during preprogrammed roll maneuvers at several constant-velocity flight conditions. These five-degree-of-freedom checks have been found to be most adequate in ensuring correct stability characteristics throughout the flight regime. Very good accuracy is obtainable with the analog computer in stability computations, since they generally involve short time transients for which the analog is well suited.

Performance checks are also made at regular intervals, and involve comparison of IBM-computed trajectories with analog results. It is in this area that the checks become most critical. Very minor discrepancies in the computer operation can cause excessive errors in performance checks, primarily in altitude and range. This is because of the long time computations involved (from 5 to 20 minutes, depending on the condition) and the resulting susceptibility of the solution to computer drifts and noise, and to nonlinear servo characteristics. It has been possible, however, to recognize the source of these errors and, by proper computer checks, to obtain the accuracy necessary for the X-15 performance problem. The accuracy usually obtained in computing such parameters as range, altitude, and velocity over a 20-minute simulated flight is within 2 percent.

Final proof of the validity of the simulation is, of course, demonstrated in how well it compares with actual flight results. The pilots who have flown the X-15 consider the simulator to be a very close simulation of the actual airplane flight characteristics. In fact, the simulator has become an integral part of the pilot's flight preparation program as a result of the pilot's own acceptance of the accuracy and value of the simulator. Figure 12 shows a performance comparison of XLR-11 maximum-speed flight with the analog-predicted flight which was run prior to the flight. The discrepancies noted are, to a great extent, the result of atmospheric variations, i.e., nonstandard days and wind shears. Figure 13 shows the same comparison for the maximum-altitude flight with the XLR-11 engine. During all speed and altitude buildup flights, simulator-predicted performance has been matched to within .1 Mach and 3000 feet altitude. The stability characteristics of the X-15 have also been adequately predicted on the simulator, although it is in this area that minor discrepancies in wind tunnel data reflect the largest differences in simulator matches. The simulation is used, in this regard, to determine the variation in stability derivatives necessary to obtain exact flight data matches, and in this manner the validity of wind tunnel data is established.

Figure 14 shows a simulator comparison of the data obtained during the landing of the first X-15 glide flight. This demonstrates the validity of the simulation in duplicating the low-speed dynamics and performance of the airplane, but at the same time, it may be used to indicate an area in which the simulation was proven to be deficient. The pitching oscillation experienced during the landing flareout, although duplicated almost perfectly on the simulator after the flight, was not predicted from simulator experience prior to flight. The problem resulted from a higher level of control sensitivity associated with the side-arm controller than was expected. This characteristic was not predicted on the simulator because of lack of critical cues, such as the visual horizon and the actual motion of the airplane, during the landing simulation.

It is of interest to compare the X-15 simulation requirements with those of currently projected manual space vehicles, and to consider some of the expected problem areas in view of X-15 experience. In the X-15 program, the major simulation requirements have been described as being related to control system development and testing, stability evaluation, pilot task studies, mission planning, and pilot training. These requirements result primarily from the fact that the vehicle utilizes the man for command and control, and that it is designed to fly at conditions far advanced over any encountered at the time of the vehicle design. The extension of this reasoning to manned space systems is obvious. The system development and testing requirements for these advanced vehicles will far exceed those of the X-15 because of the greater number of systems requiring integration and pilot evaluation and because of their increased sophistication and complexity. The importance of early simulator evaluation of subsystems critical to the space mission is demonstrated by the advanced system programs which have been accomplished, or are being planned, on the X-15 simulation. The stability and control characteristics of aerospace vehicles, particularly during re-entry, will require simulator analysis and pilot evaluation over a much greater range of flight conditions than for the X-15. The effects of the expanded flight envelope will also be reflected in increased simulation requirements in areas of human engineering, mission planning, and pilot training.

Even these very general considerations leave no doubt that simulation will play an even more important part in space vehicle development and testing than was required on the X-15. Consider next the problems which may be expected in providing this required simulation capability. The critical problem areas which have been encountered in the X-15 simulation have already been described as being related primarily to performance. These resulted from long time computation requirements in which computer drift and accuracy limitations become critical. It is apparent that these problems will become more serious in space missions simulation because of the larger mission time. The ranges of important variables which must be mechanized for a typical aerospace configuration will be, in many cases, greater than those required in the X-15 simulation and will create additional scale-factoring problems. Typical of these are velocity, Mach number, and altitude. In some instances, however, the X-15 requirements may be the more severe. For example, the range of dynamic pressure for which the X-15 is designed (0 to 2500 psf) is much greater than will be required in winged re-entry vehicles. The rates of change of dynamic pressure in the X-15 mission are even more significant. Figure 15 shows dynamic pressure rate of change during re-entry as a function of re-entry angle. It is seen that the steep re-entry angles required for the X-15 result in much more severe dynamic pressure changes than are experienced by orbital re-entry vehicles.

Space simulation requirements will allow particular phases of the complete mission to be mechanized individually in order to limit the range of variables and to minimize the computation time required. For instance, simulation of the boost, orbit, and re-entry phases of a complete orbital mission in a single mechanization becomes impractical by current analog techniques. However, if each phase is mechanized individually, with appropriate scale factoring, satisfactory results may be obtained for most applications. For example, the major portion of the orbital re-entry problem may be simulated at altitudes above 100,000 feet. Since the X-15 simulation adequately duplicates air density from sea level to 250,000 feet, the requirements for simulation of air density for an orbital re-entry problem are no more severe than for the X-15.

Some preliminary studies have been conducted utilizing conventional analog techniques to mechanize various aerospace missions. The vehicles simulated were an orbital re-entry system and a lunar return vehicle. The X-15 flight simulator was used to allow pilot participation in analysis of such problems as high-angle-of-attack stability during re-entry and flight path control during aerodynamic deceleration from lunar return velocities. Analog data showing a piloted lunar return maneuver are shown in figure 16. As the vehicle approaches the earth at approximately 36,000 feet per second along an optimum flight path which has been established many hours earlier, the pilot rolls 180 degrees and, by utilizing aerodynamic lift, maintains a constant altitude as the velocity slows to suborbital. In this application, the analog simulation was found to be quite adequate in establishing pilot capability and control techniques, since the critical portion of the problem (namely, the establishment of the proper approach trajectory) was assumed to have already been accomplished. Also, the maneuver considered required relatively high aerodynamic forces at velocities appreciably greater than orbital, so that the differences between gravity and aerodynamic effects were not critical. Figure 17 shows the effects of a 0.1 percent error in either of the components of acceleration during a simulated re-entry from a near-circular orbit in which aerodynamic drag is used to decelerate the vehicle at perigee. Here the effects on performance during re-entry are seen to be quite severe due to the near-equal values of gravity and centrifugal acceleration. It was apparent from these studies that the analog mechanization provided adequate simulation of both dynamic characteristics and performance, provided the computation time was minimized and good performance was not required at very-near-orbital velocities.

To summarize, it would be well to reiterate that the X-15 flight simulator, as it exists today, is the result of design requirements as projected early in the program. The requirement to include the pilot in the loop as an integral part of the systems development and testing provided the major justification for the detailed duplication of displays and flight controls, and for the extensive six-degree-of-freedom analog simulation of the airplane motion. Since the application for which the simulator has been utilized in the latter phases of the program (namely, mission planning and pilot training) were not clear cut in the beginning, we were fortunate that early requirements dictated the simulation approach which was pursued. It has been indicated that the X-15 simulation requirements were similar in many respects to those which may be expected in simulation of more advanced vehicles, and that the problems which were encountered are definite indication of the difficulties which may be expected. Consequently, it appears that the X-15 simulation requirements, techniques, and problems should prove helpful in future programs, both in early definition of the overall simulation program to be followed and in determining the best approach to accomplish the program.

Symbols

b	Wingspan (ft)
$\bar{C}$	Mean aerodynamic chord (ft)
C_c	Chord force (nondimensional coefficient)
C_l	Rolling moment (nondimensional coefficient)
C_m	Pitching moment (nondimensional coefficient)
C_n	Yawing moment (nondimensional coefficient)
C_n	Normal force (nondimensional coefficient)
C_y	Side force (nondimensional coefficient)
G	Acceleration of gravity (ft/sec^2)
h	Altitude (ft)
I_{xx}	Moment of inertia about X-body axis (slug-ft^2)
I_{yy}	Moment of inertia about Y-body axis (slug-ft^2)
I_{zz}	Moment of inertia about Z-body axis (slug-ft^2)
l_T	Tail length (ft)
m	Mass (slugs)
M	Mach number
p	Roll rate
q	Pitch rate
q_o	Dynamic pressure (lb/ft^2)
r	Yaw rate
S	Wing area
V_o	Velocity of the center of gravity (ft/sec)
V_x	Velocity component along X-body axis (ft/sec)
V_y	Velocity component along Y-body axis (ft/sec)
V_z	Velocity component along Z-body axis (ft/sec)
α	Angle of attack
β	Angle of sideslip
δ'	Differential horizontal stabilizer deflection, roll control
δ_H	Horizontal stabilizer deflection, pitch control
δ_V	Vertical stabilizer deflection, yaw control
Φ	Euler angle of axial rotation, roll
Θ	Euler angle of elevation, pitch
Ψ	Euler angle of azimuth, yaw

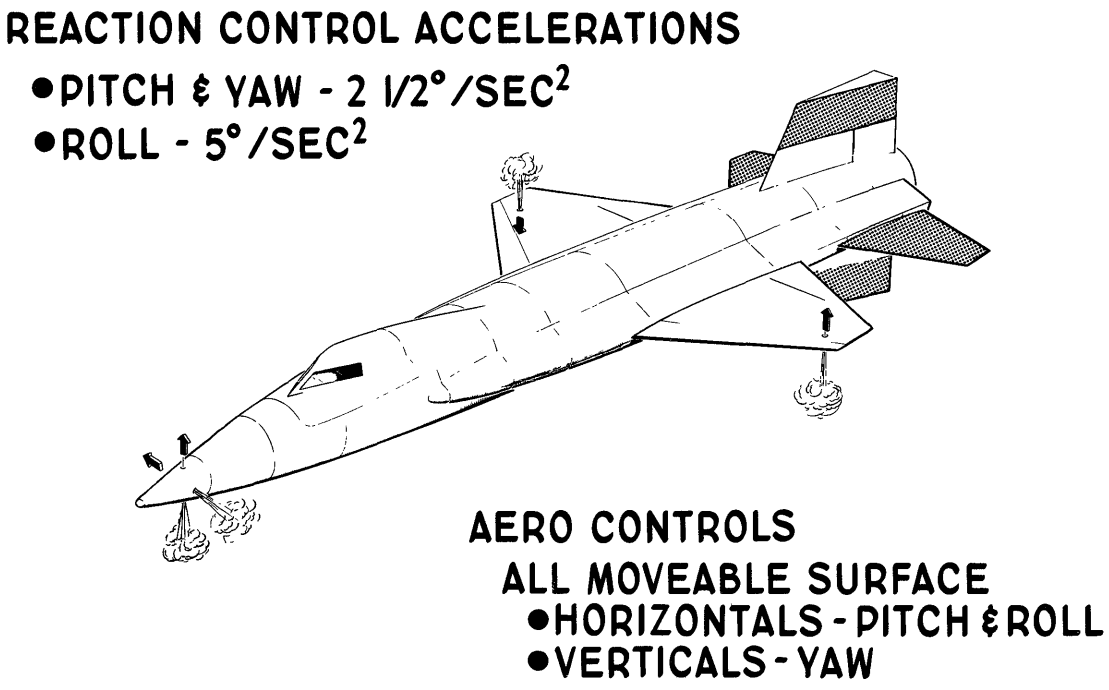

REACTION CONTROL ACCELERATIONS
•PITCH & YAW - 2 1/2°/SEC^2
•ROLL - 5°/SEC^2
AERO CONTROLS
ALL MOVEABLE SURFACE
•HORIZONTALS - PITCH & ROLL
•VERTICALS - YAW

Figure 1. Flight Controls

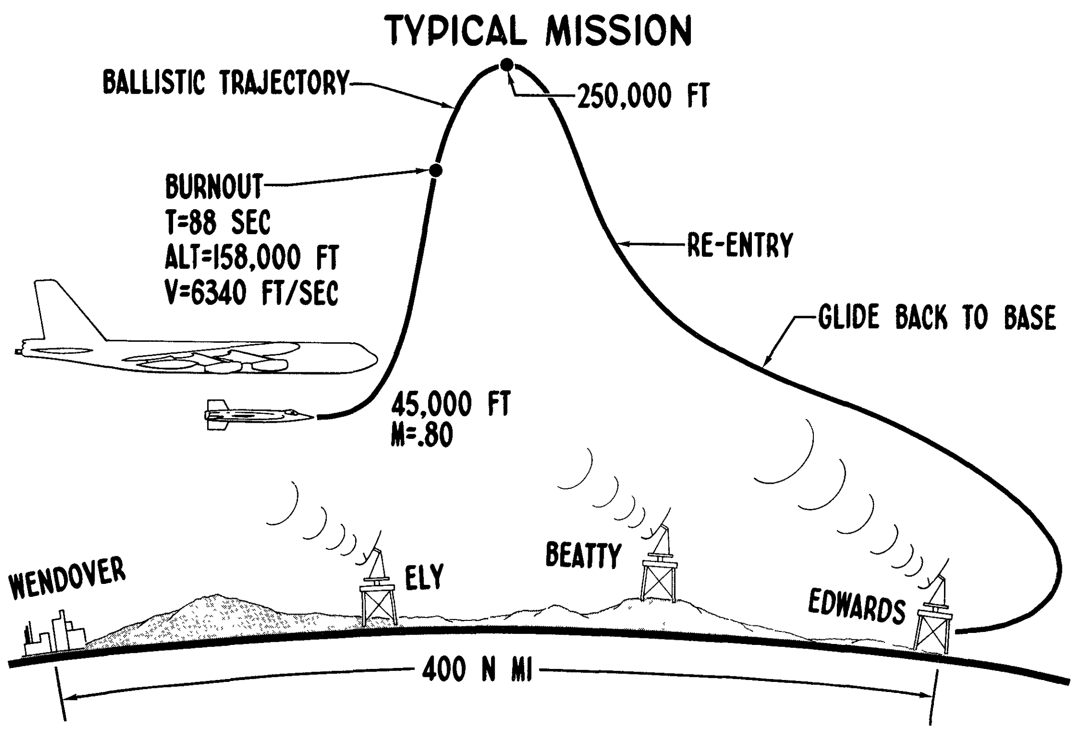


Figure 2. X-15 Research System

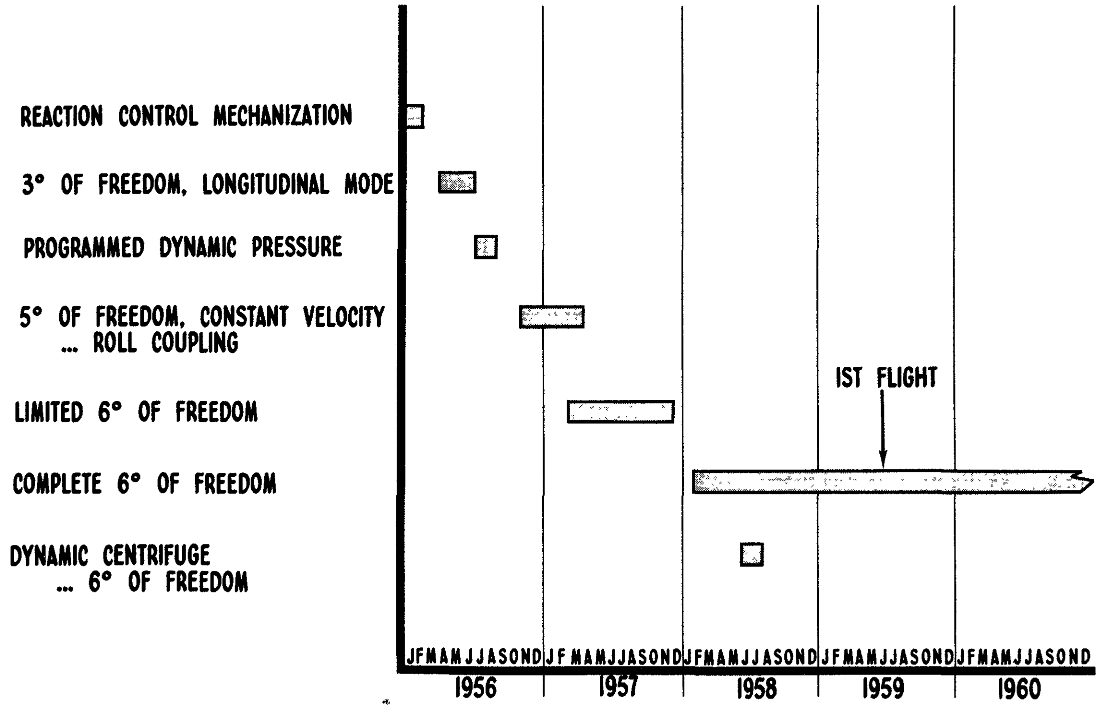


Figure 3. Flight Simulation Summary

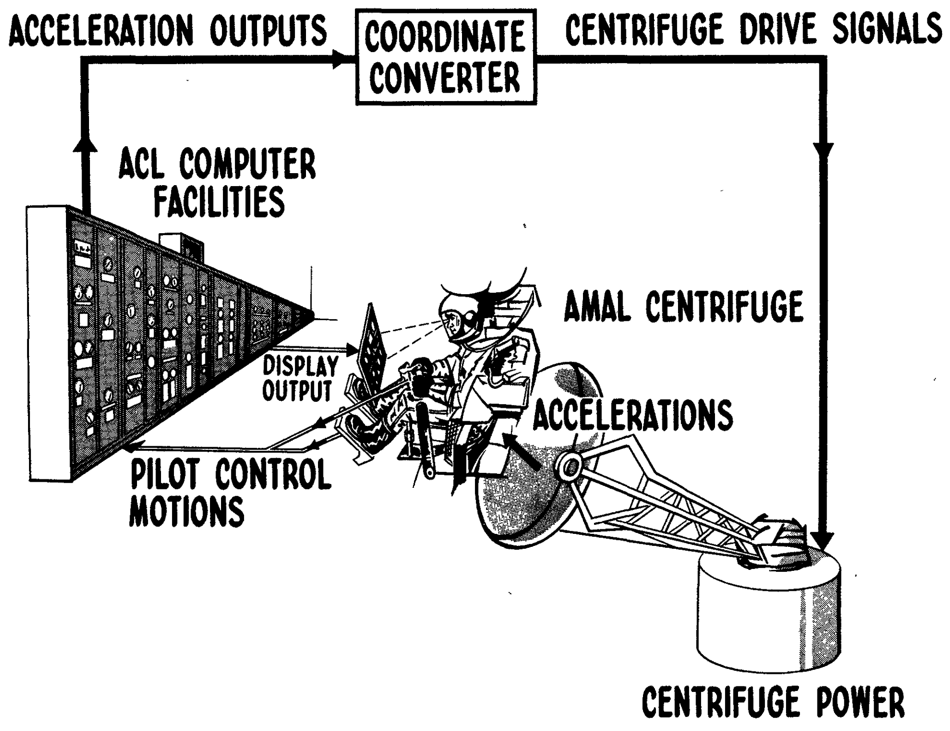


Figure 4. Dynamic Simulation

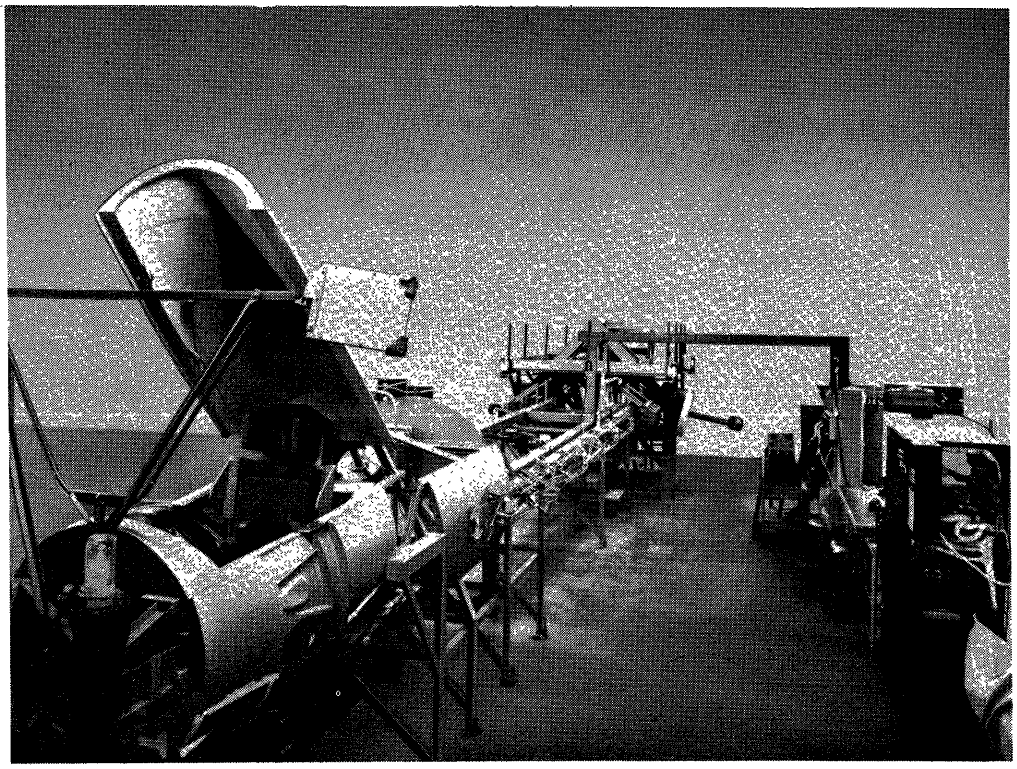


Figure 5. Flight Control Simulator

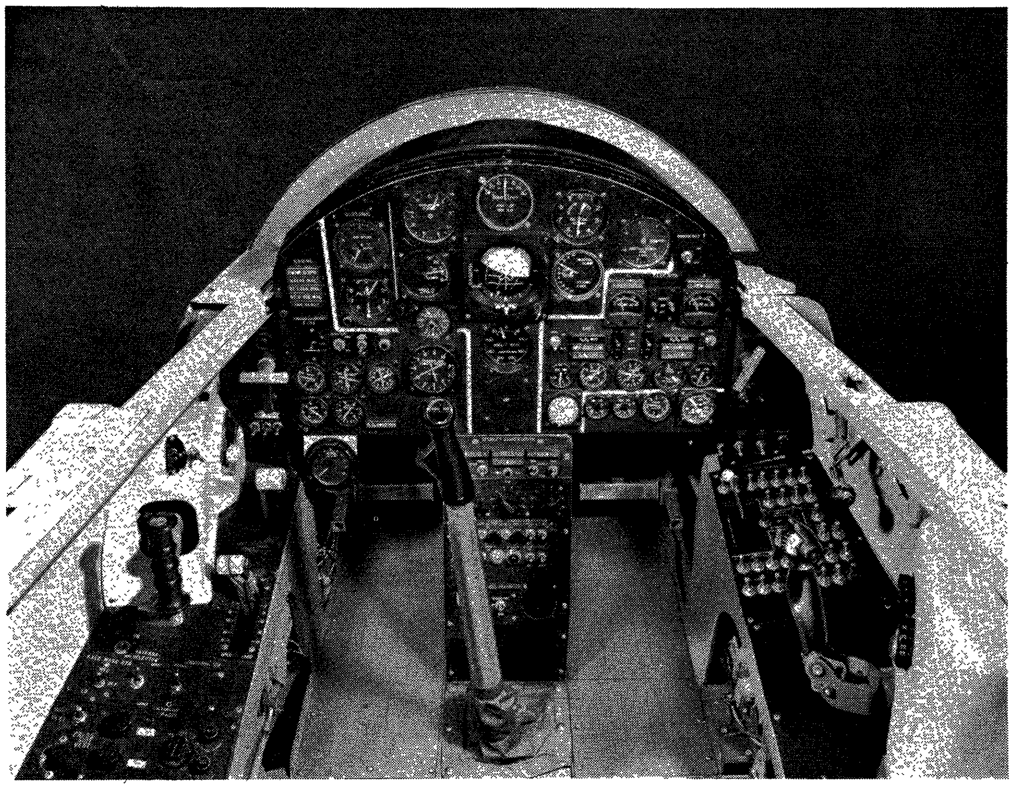


Figure 6. Simulator Cockpit

$$\dot{V}_x = g \sin\Theta + \frac{T - q_0 s C_c}{m} - V_z q + V_y r$$

$$\dot{V}_y = g \cos\Theta \sin\Phi + \frac{q_0 S}{m}\left(C_{y_\beta}\beta - 1.6\, C_{n_{\delta_V}}\delta_V - 1.5\, C_{n_{\delta'}}\delta'\right) - V_x r + V_z p$$

$$\dot{V}_z = g \cos\Theta \cos\Phi - \frac{q_0 S}{m}\left(C_{N_{\delta_H = 0}} - \frac{\bar{c}}{\ell_T} C_{m_{\delta_H}}\delta_H\right) - V_y p + V_x q$$

$$\dot{p} = \frac{q_0 s b}{I_{xx}}\left(C_{\ell_\beta}\beta + C_{\ell_{\delta_V}}\delta_V + C_{\ell_{\delta'}}\delta' + \frac{b}{2V} C_{\ell_p} p\right) + \frac{I_{yy} - I_{zz}}{I_{xx}} q r$$

$$\dot{q} = \frac{q_0 s \bar{c}}{I_{yy}}\left(C_{m_{\delta_{H=0}}} + C_{m_{\delta_H}}\delta_H + \frac{\bar{c}}{2V} C_{m_{q_e}} q\right) + \frac{I_{zz} - I_{xx}}{I_{yy}} p r$$

$$\dot{r} = \frac{q_0 s b}{I_{zz}}\left(C_{n_\beta}\beta + C_{n_{\delta_V}}\delta_V + C_{n_{\delta'}}\delta' + \frac{b}{2V} C_{n_r} r + \frac{I_{xx} - I_{yy}}{I_{zz}} p q\right.$$

$$\dot{\Theta} = q \cos\Phi - r \sin\Phi \qquad\qquad V_0^2 = V_x^2 + V_y^2 + V_z^2$$

$$\dot{\Phi} = p + \dot{\Psi} \sin\Theta \qquad\qquad \alpha = \sin^{-1}\frac{V_x}{V_0 \cos\beta}$$

$$\dot{\Psi} = \frac{q \sin\Phi + r \cos\Phi}{\cos\Theta} \qquad\qquad \beta = \sin^{-1}\frac{V_y}{V_0}$$

$$\dot{x} = \left[V_x \cos\Theta + (V_y \sin\Phi + V_z \cos\Phi)\sin\Theta\right]\cos\Psi + \left[V_z \sin\Phi - V_y \cos\Phi\right]\sin\Psi$$

$$\dot{y} = \left[V_x \cos\Theta + (V_y \sin\Phi + V_z \cos\Phi)\sin\Theta\right]\sin\Psi - \left[V_z \sin\Phi - V_y \cos\Phi\right]\cos\Psi$$

$$\dot{h} = V_x \sin\Theta - V_y \cos\Theta \sin\Phi - V_z \cos\Theta \cos\Phi$$

$$g = 32.17 - k_1 h - k_2 \dot{x}^2$$

Figure 7. Six-degree-of-freedom Equations of Motion

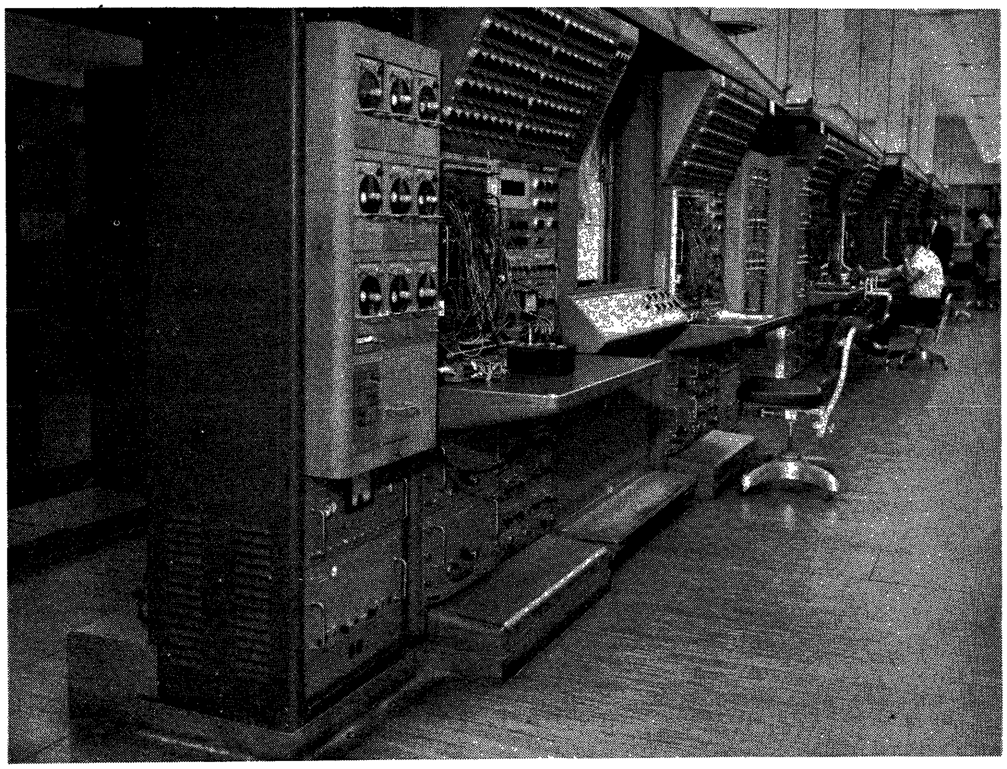


Figure 8. Analog Simulation

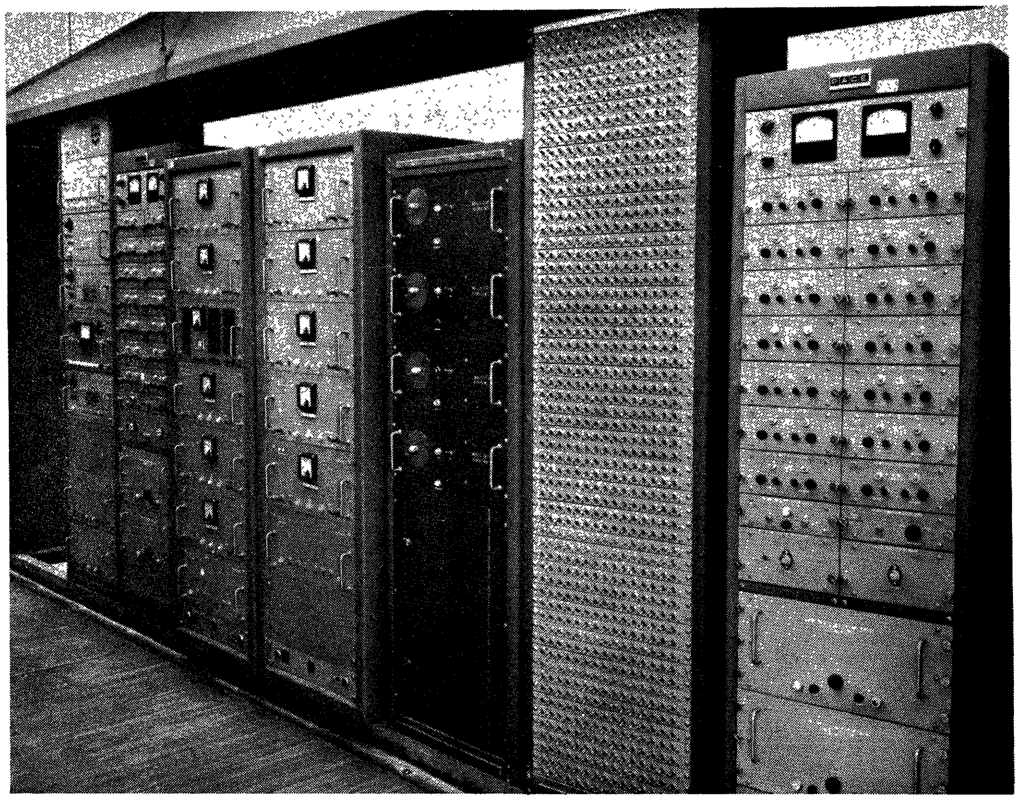


Figure 9. Nonlinear Computing Equipment

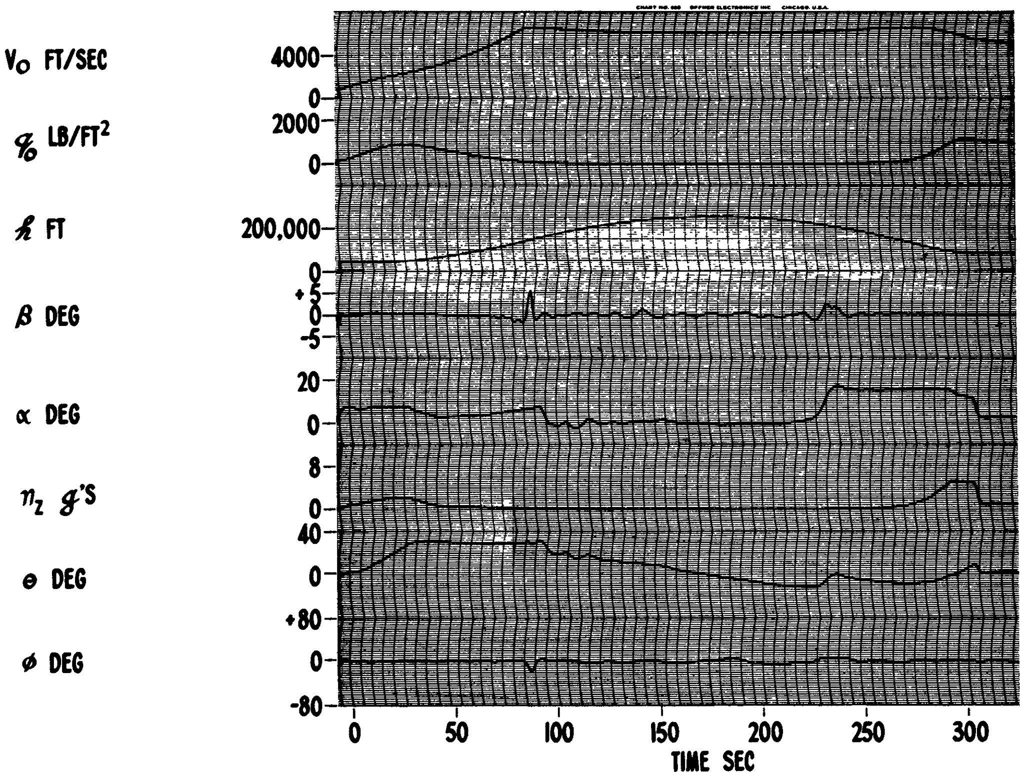


Figure 10. Design Altitude Mission

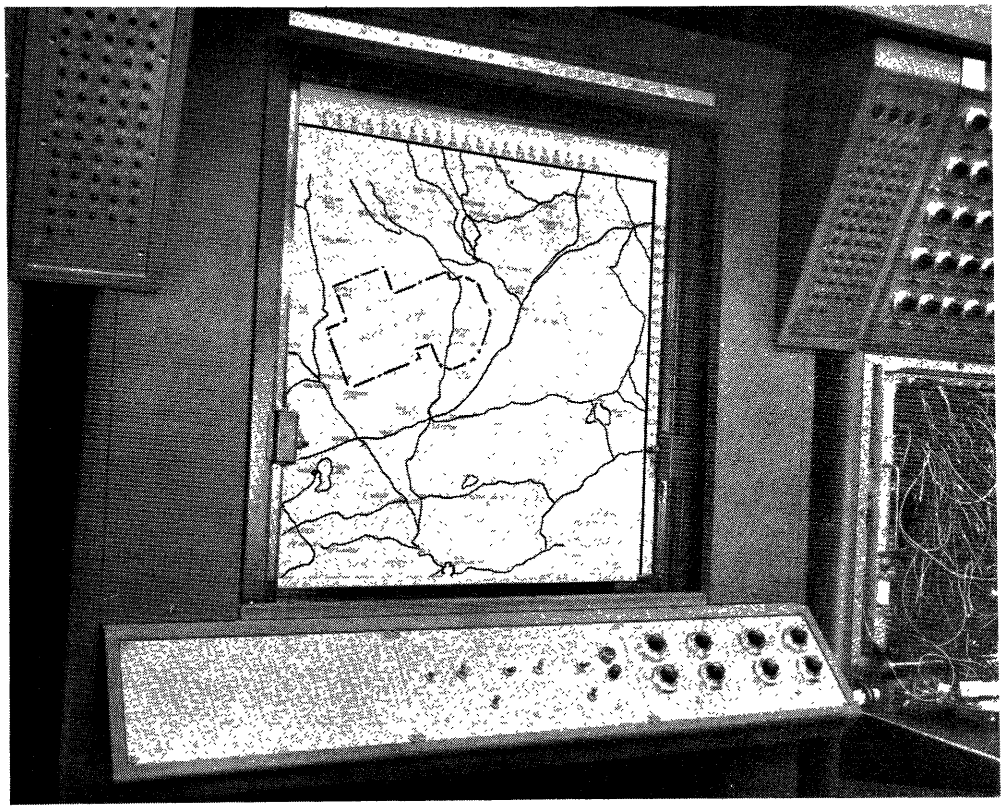


Figure 11. Variplotter for Simulated High-range Plotting

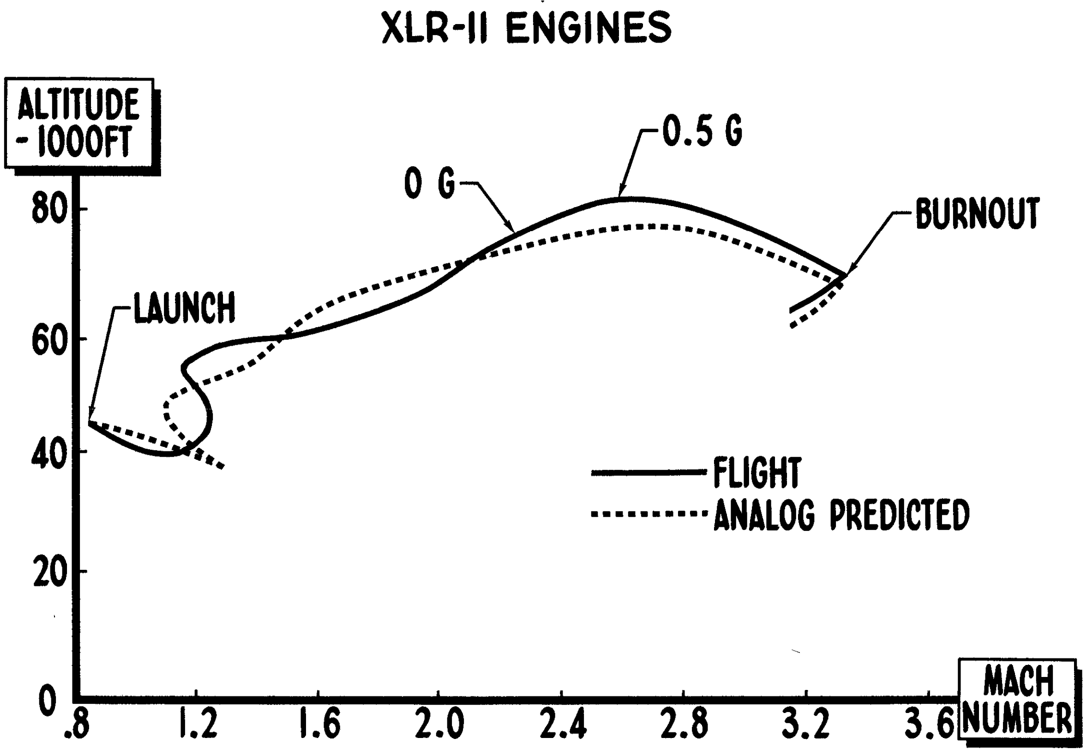


Figure 12. X-15 Maximum-speed Flight

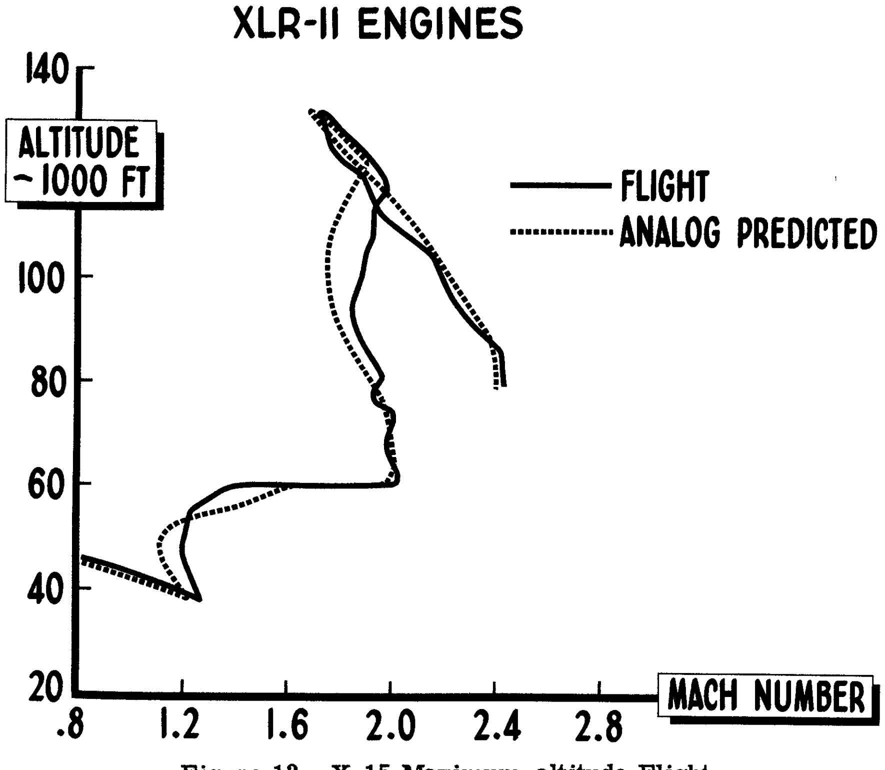


Figure 13. X-15 Maximum-altitude Flight

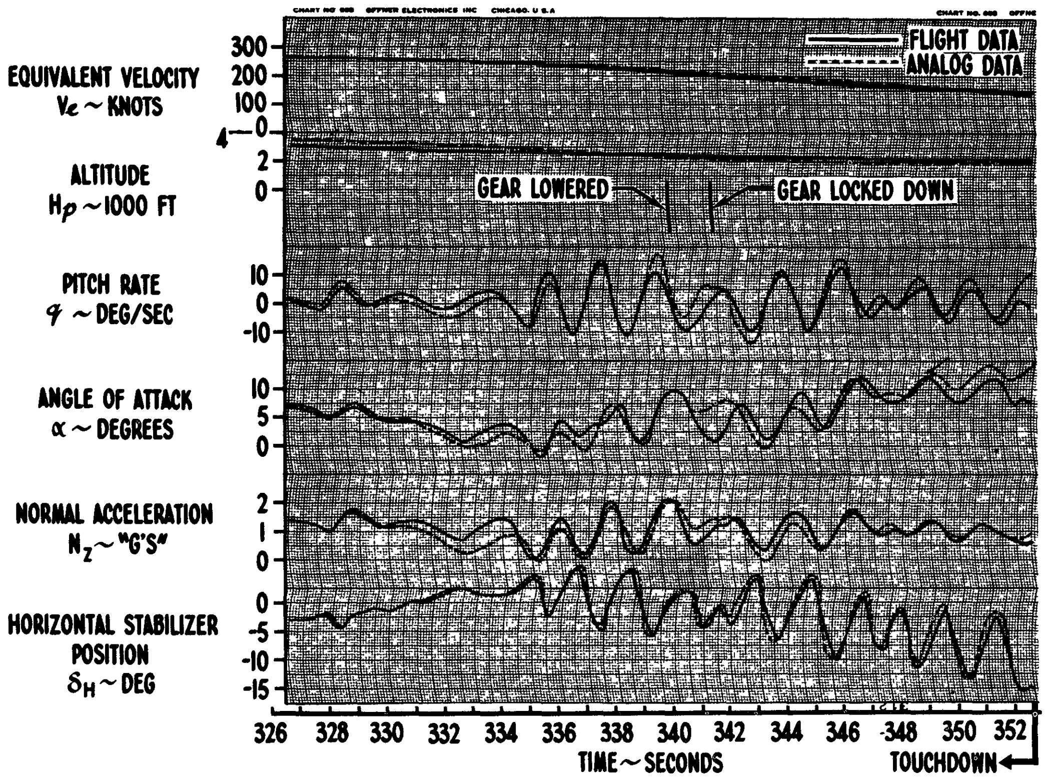


Figure 14. First Glide Flight Landing Flareout

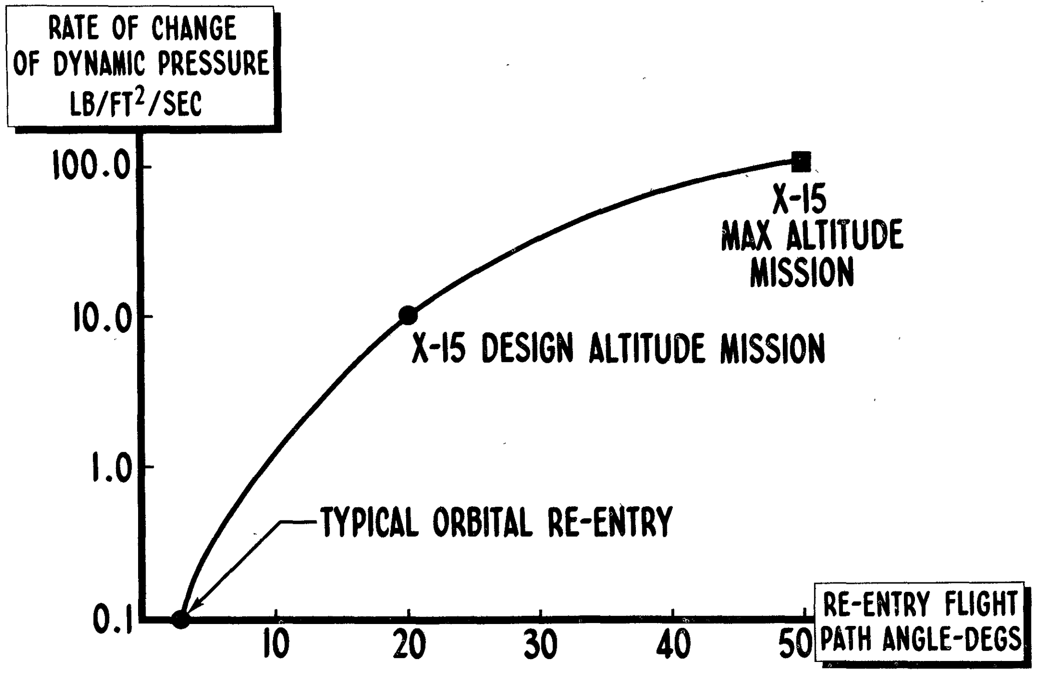


Figure 15. Re-entry Dynamic Pressure Variation

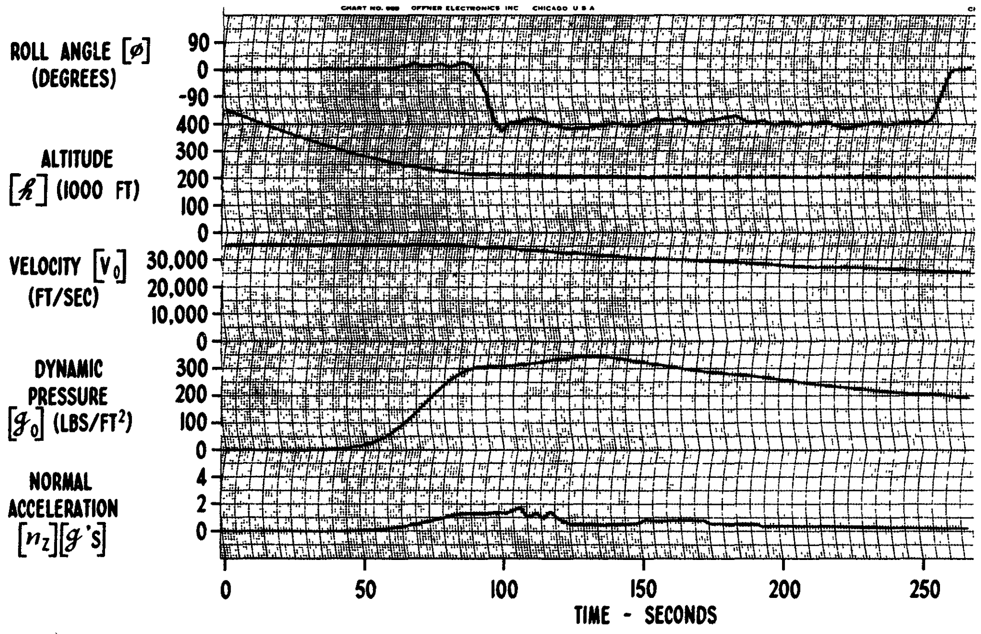


Figure 16. Lunar Vehicle Re-entry

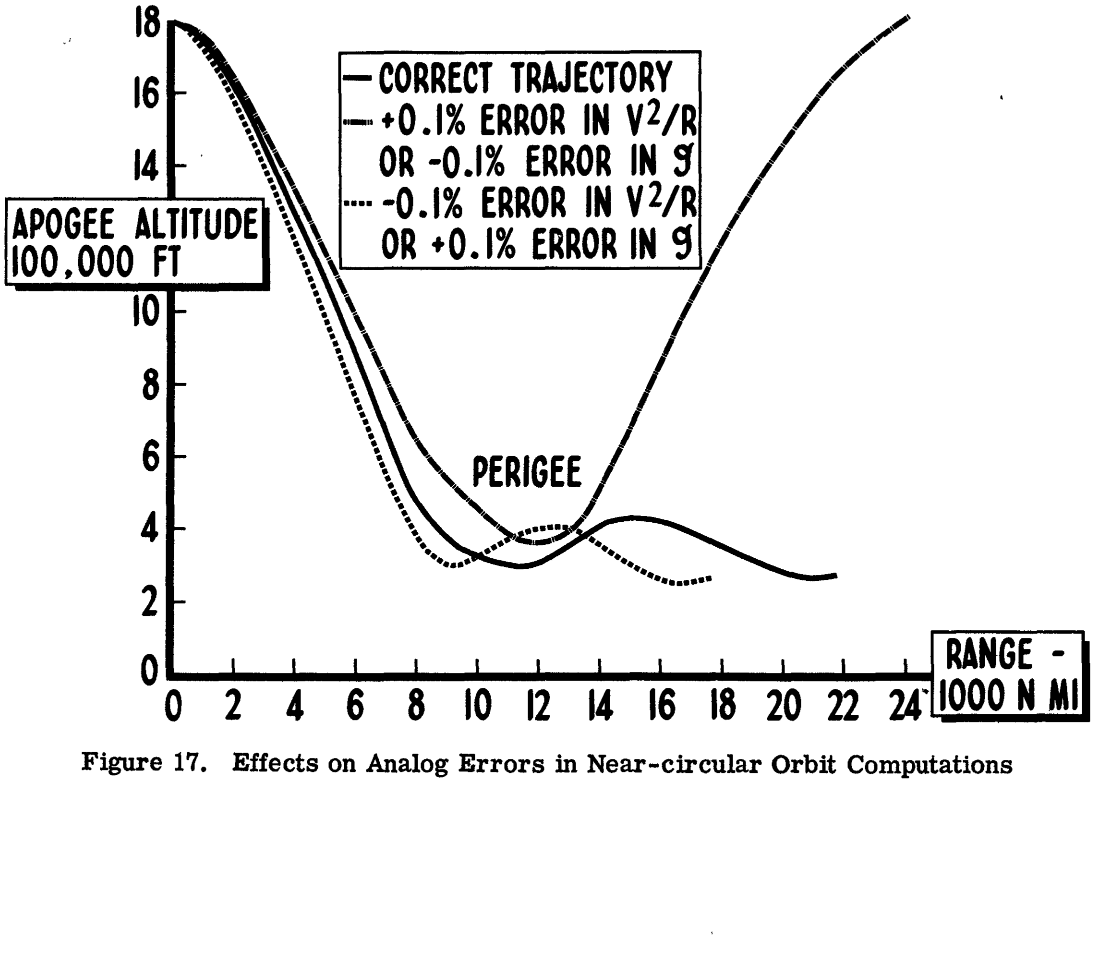


Figure 17. Effects on Analog Errors in Near-circular Orbit Computations

ANALOG - DIGITAL HYBRID COMPUTERS IN SIMULATION WITH HUMANS AND HARDWARE

Owen F. Thomas
U. S. Naval Ordnance Test Station
Pasadena, California

The U. S. Naval Ordnance Test Station, Pasadena, operates an analog simulation center for testing antisubmarine torpedoes and the associated fire-control equipment. This simulation incorporates a real torpedo on a flight table, a real fire-control computer, and real fire-control personnel. The simulation must be run in real time in order to place the proper requirements on the components being tested, for example, the human operators must not be allowed any extra decision-making time. Now, it is intended to incorporate a digital computer into the simulation. This leads to the general system to be considered here -- an analog - digital hybrid computer required to work in real time with humans and hardware. Certain problem areas are defined and some techniques of solution are presented.

The Problem

The problem is to test a physical device.

Mathematical Simulation

In this type of testing, a mathematical model is formed for each physical device involved and these models are programmed on a computer. It is then possible to consider that the computer is acting as if it were the physical devices that are modeled. While executing the program, the computer produces graphs as a function of time, or lists of times along with quantities like velocity, position, and fuel expended to describe the conditions of the physical devices at various times. In such a solution, clock time is of importance only in determining the cost of computation. It is of secondary importance whether 1/10-second time intervals on the printed page were actually printed at 1/10-second intervals by the operator's wrist watch. Mathematical models of this sort are interesting, but if a human, or hardware, is involved in the solution without being modeled, then the model must move in clock time. Furthermore, this type of model fails to convince the hard-boiled engineer who knows that there is many a slip between the equations and a working device.

Simulation Robot

Characteristics. The simulation center which NOTS operates uses mathematical models when they are unavoidable, but allows a real device to perform before the eyes and instruments of an engineer wherever possible. It is not sufficient here for the computer to print that the torpedo is turning at a rate of 20 degrees per second during a search scan. The torpedo on the flight table must actually rotate at 20 degrees per second. The essential feature of the NOTS simulation is that mathematical answers cause movement of some sort or produce information for human operators. Things that move and talk to humans have been called Robots. Therefore, this facility is defined as a Simulation Robot. The computing equipment is called a Robot Brain. The Brain must receive stimuli, think about them, and cause an action, while recording certain quantities of interest. The Brain is adequate only if its response is quick enough and accurate enough. If a digital computer is part of the Brain, it is often necessary to sacrifice accuracy in order to gain speed.

The device which an engineer has built and wants to test is also a Robot. It is intended to observe the real world in some way and perform some sort of action in response. In the real world, these Robots go dashing about with much noise and violence, often destroying themselves, and generally they are very hard to observe. The Simulation Robot is meant to cradle them safely in its arms and make them think they are in the real world while the engineer watches them, as shown in Fig. 1. The Simulation Robot regards the engineer's device as merely a Test Robot, which is what it will be called from here on. The Test Robot might even be a man, for example a fire-control officer, and there may be multiple Test Robots.

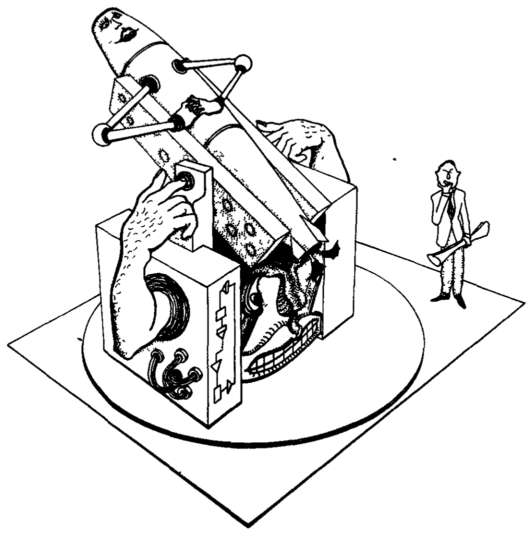


Figure 1

Now it is clear what the Simulation Robot must do. It must duplicate the environment of the Test Robots. In greatest generality, it would be like the whole wide world. In special cases, it must at least be enough like a small part of the world so that the Test Robot will not know the difference. The engineer might also like to check out a small part of a Test Robot before it is all completed, in which case, the Simulation Robot is required to behave like the missing parts. The engineer might also like some assistance from the Simulation Robot in the area of assigning test jobs to the Test Robot and in evaluating performance. He might even like the Simulation Robot to decide upon new test jobs as a result of the previous performance. The Brain of the Simulation Robot should be composed of whatever computing elements can be used for these jobs. Remember that the most important job of the Simulation Robot is that it be like at least part of the environment. More specifically, it must be like the parts of the environment that are important to the Test Robots.

Inertial Force Simulation. Simulation at NOTS has included inertial forces on the torpedo from the environment for many years. When the Test Robot actuates a control surface, the information goes to an analog part of the Simulation Robot Brain that solves the hydrodynamic motion equations and controls the flight table that holds the torpedo. This results in some inertial forces being applied just as they would be in the ocean. The forces due to lateral acceleration are not simulated because the flight table cannot move laterally. This has caused no difficulty because the Test Robots have not been sensitive to lateral accelerations. It is not planned to use digital equipment in this part of the Brain, as it would require analog-digital conversion or a digitally controlled flight table or both.

Environmental Response Simulation. When a Test Robot interrogates the environment, the Simulation Robot must fake a response. For many types of interrogation, the response is mathematically determined by a solution of the wave equation.of mathematical physics. The boundary conditions for this solution are time-varying and the medium has spatial and time variability. What is needed is a computer to solve these equations with the same speed as a response from the real environment. It is well-known that present computers cannot do this. The present effort is to simplify the mathematical description of acoustic echoes to a point where they can be generated by the Brain.

Human Factors. Humans do not interrogate the environment in this simulation; instead they operate certain controls on the Test Robots or on consoles concerned with Test Robot deployment. Therefore, it is of no concern at present that there be direct communication between humans and the Simulation Robot Brain during a simulation run. Of course, human operators are involved in starting and stopping the Simulation Robot, and in programming it, or in giving it the basic education which it must have.

Solution Techniques

Acoustic Echo Simulation, First Method

The Test Robot occasionally transmits a pulse of sinusoidal pressure waves and observes the return. This particular case of environmental solution will be the first to be incorporated in the digital part of the Simulation Robot Brain. The return can be obtained by solving the wave equation, but the initial effort will be based on a simplification. It is assumed that the echo is a sum of pulses just like the one transmitted but delayed by varying amounts of time. For the first model, it is also assumed that there is no doppler shift, so the solution will be in the form of a sum of pulsed sine waves, all at the same frequency.

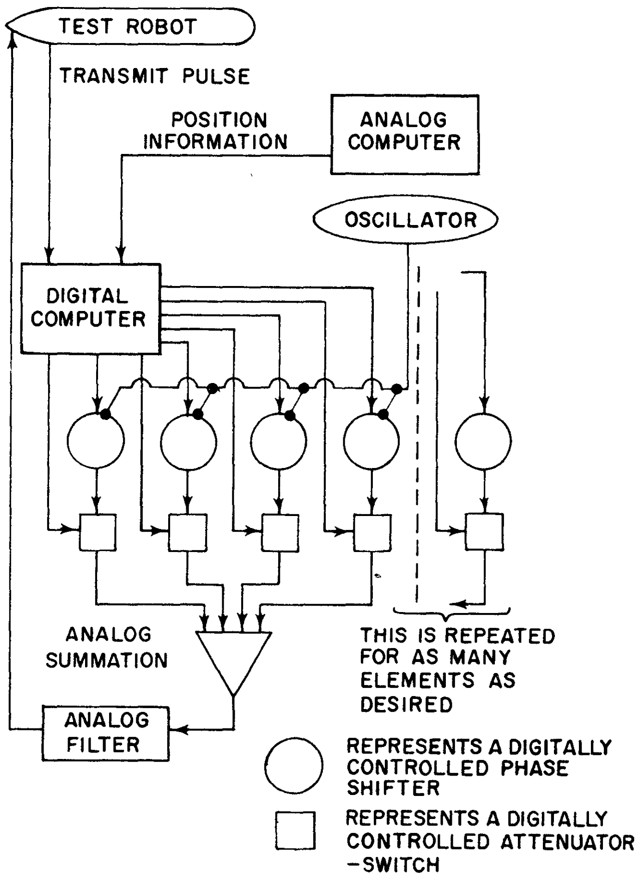

TEST ROBOT
TRANSMIT PULSE
POSITION INFORMATION
ANALOG COMPUTER
OSCILLATOR
DIGITAL COMPUTER
ANALOG SUMMATION
ANALOG FILTER
THIS IS REPEATED FOR AS MANY ELEMENTS AS DESIRED
REPRESENTS A DIGITALLY CONTROLLED PHASE SHIFTER
REPRESENTS A DIGITALLY CONTROLLED ATTENUATOR -SWITCH

Figure 2

Figure 2 is a block diagram of the components involved in the solution. A summation of sine waves is not a simple arithmetic operation like a summation of numbers. In this solution, the summation is shown being performed by an analog summing amplifier. The various pulsed sine waves are generated by analog phase shifters and attenuator-switches, controlled by the digital computer. The digital computer has the job of computing the phase, amplitude, and time of each component of the echo and controlling the analog elements. It computes this information on the basis of position data obtained from the analog computer via analog to digital converters. Note that the Test Robot does not actually transmit into the environment; it sends a pulse to the digital computer. Note also that the Test Robot does not observe the environment; it receives the analog solution fed through an analog filter with the same transfer characteristics as its hydrophone.

Some may wonder why it is necessary to go to such extreme measures and it must be stated that the problem has been simplified for this presentation. Actually, the signal is more complicated and the Test Robot is doing a meticulous job of signal processing on the sine wave sum which it receives. This simple example still serves for discussion of techniques. One thing of interest is the filter. It is an analog device; the digital computer would take too long to do the filtering job. Further, it is significant that the digital computer is not outputting the sine wave of interest. It is too slow. It is used instead to control an analog oscillator. Note also that the digital computer is not being asked to solve the real-time problem of the wave equation in the real world. Instead, it is working with a simplified model that allows computing to be done during a time which is considered as transport delay while the transmitted signal travels to the target and returns. This is possible for sonar where the velocity of propagation is relatively slow. There is some difficulty in doing this for radar unless the range is quite large.

This system is capable of simple extension to the case where each component in the echo has its own doppler shift. It would merely be necessary to use a number of oscillators with frequency controlled by the digital computer. However, the cost of this system is high because there are many digitally controlled analog devices; a cheaper, slower, and less versatile system will therefore be considered.

Acoustic Echo Simulation, Second Method

Figure 3 is a block diagram of a simplified system. There is now only one oscillator with phase and amplitude controls. The rest of the system is unchanged, so that the output of the single controlled oscillator must be the same as the output from the analog summation amplifier in Fig. 2. This summation is mathematically equivalent to a vector summation which must now be performed in the digital computer. The digital output quantities are the phase angle and amplitude of the sum vector. Each time a new sine wave echo component enters the sum, the digital computer must go through at least sine, cosine, square root, and arctangent calculations. This means that it will be slower in response.

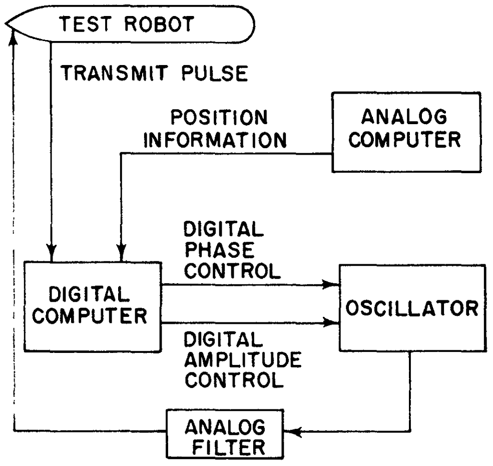


Figure 3

Because there is a limited amount of time available, this system cannot consider as many components as can the system of Fig. 2. It is, therefore, not as accurate in the available time.

Communications Between Computers

Description by Sine Wave Components. The systems of echo simulation considered communicate a simplified description of the echo, namely, the amplitudes and phases of sine waves. These are similar to Fourier coefficients, except that the coefficients are functions of time. This type of thinking is also applicable to input problems. If it is known that an input signal is a sine wave, then an input unit should be built to extract its amplitude, phase, and frequency, and input these three numbers. This has an obvious extension for signals consisting of more than one frequency.

Description by Derivatives. Some signals are easily described by their derivatives. In analog computations, this is almost always the case. Figure 4 is an example. It results in a parabolic extrapolation of the conditions at time t_0. At t_0, the digital computer establishes initial conditions and input to an analog extrapolator composed of two integrators. This is done by an output of $x(t_0)$, $\dot{x}(t_0)$, and $\ddot{x}(t_0)$ as shown. The value $x(t)$ will then be a parabolic extrapolation for all times t. If the second derivative does not change, this will be exact; if it does change, the digital computer can reset the output extrapolator periodically, resulting in a parabolic segment approximation. A segmented straight-line approximation could be done by eliminating one integrator and setting only $x(t_0)$ and $\dot{x}(t_0)$. This can be increased in complexity to give any desired degree of approximation. This type of trick could also be done on input because a similar prediction can be made in the digital computer on the basis of analog input signals of $x(t_0)$ and its derivatives.

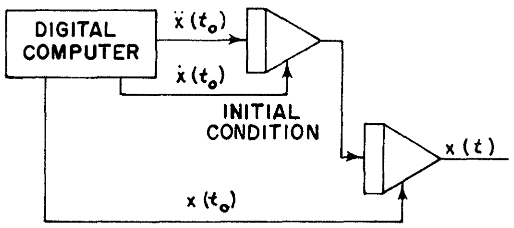


Figure 4

Communication Codes. At the present time, general purpose digital computers will accept only digital stimuli and will respond only in digital form. The problem of connection to the real world is left up to the ingenuity of special equipment designers who usually surround the digital computer with analog-digital and digital-analog converters and then use familiar analog transducers or analog-controlled manipulators for input and output. It should be kept in mind that there are many forms of encoders for analog-digital conversion, and there are digitally controlled manipulators; a system is not limited to the one type of conversion.

It should also be noted that the communication line between Test Robot and Simulation Robot is not invariant. For example, the Test Robot may not actually radiate energy; the load may be artificial and the Simulation Robot stimulus may be picked up from the controls of the Test Robot. In a similar manner, the Simulation Robot may introduce a signal into the Test Robot somewhere after a sensing element. If the Test Robot happens to have a part digital brain, this could be very convenient. In cases of this sort, part of the Test Robot is not functioning and the engineer must decide if there is an essential test being bypassed.

The Future of Communication. In some bright future, it would be advantageous to have the digital equivalent of analog quantities automatically available in some memory locations and to have some memory locations control manipulators directly. All present computers require main program steps to cause data transfer. The block diagram of a desirable system is shown in Fig. 5. The switching matrix on input is under program control and selects the analog lines that are to feed memory locations. The selected lines are converted in rotation and transmitted to consecutive memory locations. A similar switching matrix on the output connects memory locations to manipulators. It must be mentioned that there is one computer that is partly set up for this type of operation, since it has circulating memory tracks that can be written or read by a second computer of the same type. The second computer could be used for the switching matrices and transfers. The main difficulty is that the particular computer is not really fast enough.

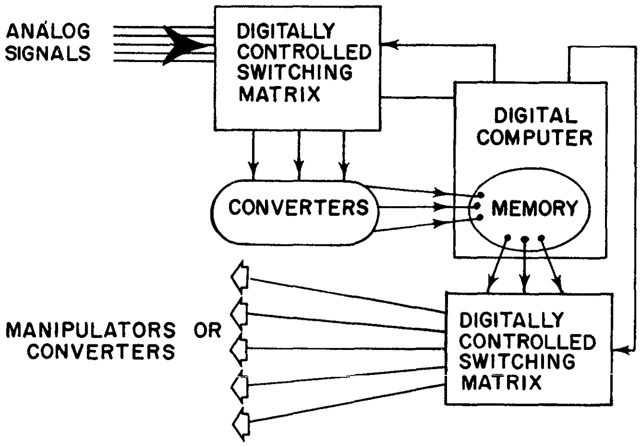

ANALOG SIGNALS
DIGITALLY CONTROLLED SWITCHING MATRIX
DIGITAL COMPUTER
CONVERTERS
MEMORY
MANIPULATORS OR CONVERTERS
DIGITALLY CONTROLLED SWITCHING MATRIX

Figure 5

Tricks With Time

In reference to the problem of giving the digital computer more time for solution, it has been suggested that the analog computer be put into hold until the digital computer is finished. This might work in some cases for a pure analog-digital hybrid, but in the present application there is no way to re-start the Test Robot with the proper initial conditions. Furthermore, this would provide the humans with too much decision-making time. Another suggestion is to rerun the problem many times, each time using recorded digital solutions and doing one more digital solution. This will not work when there is enough noise in the Test Robot to keep it from repeating its actions on reruns. It also fails when there is a Test Robot, such as a human, involved which may learn too much from the reruns.

The only way to get things done faster is to have multiple digital computers working at the same time on separate parts of the problem.

Real Time Computers

What Are They? The hare and the tortoise are both real time animals but no serious writer would claim they are the same speed. Computer salesmen seem to overlook this fact when they claim to have a real time computer. There appears to be no sales literature that approaches this problem seriously. This problem is defined below in terms familiar to analog people who do not juggle time.

The transfer characteristic and gain accuracy of an analog element describes its speed and accuracy. What is needed is a similar characteristic for digital elements. It is not immediately apparent what this characteristic is for digital elements. What is it for systems? Analog experts can predict whether a desired system characteristic is possible on the basis of the element characteristics. In the digital equipment, this is equivalent to deciding whether a program can be written to input a time function, perform a required operation on it, and output the resultant time function fast enough and with enough accuracy. A salesman who claims to have a general-purpose, real time computer should allow an arbitrary system transfer characteristic to be stated and have his computer obey it. So far, there have been none found who can do that. For now, certain limits are implied by the operations that follow.

Pulse Delay. The fastest useful job that can be conceived for a digital computer is pulse transmission with specified delay. Such a function can be done on a computer that has been considered at NOTS. It requires 65 microseconds to set the delay. The minimum delay is 10 microseconds and 10-microsecond multiples are easily available up to a total of about 1/3 second. The output pulse will be synchronized with the digital computer. If the input is not synchronized, the delay will vary because the computer will not recognize the input immediately. This indicates that, for some jobs, the entire Robot might have to be synchronized with the digital computer.

Square Wave Generation. Another job the digital computer can do rapidly is generate a square wave with controlled period and level on each half-cycle. This can be done on a computer NOTS has considered with a minimum period of 120 microseconds. The 120 microseconds are all consumed in analog-digital conversion and input. The converters require 52 microseconds and it is not possible to complete the input until 60 microseconds have elapsed. This time is required on each half-cycle for complete control. During the conversion time, all the program steps required for period control and output can be executed. The period is controllable on each half-period in steps of 10 microseconds up to a total of about 1/3 second.

Digitized Signal Delay. A slight modification of the previous program can give time variable delay to a time function. The minimum delay of 60 microseconds can be increased by multiples of 10 microseconds to about 1/3 second.

Summary. A real time computer has not been defined, but an attitude has been suggested and some bounds on speed stated.

Test Robot Performance Evaluation

Type of Brain to Use. In this area, the digital part of the Simulation Robot Brain seems clearly useful. Performance results are almost always reduced to digital form, and as such are ideal for a digital computer. These results can be tabulated for the Test Robot engineer, or they can be processed according to the engineer's rules to automatically decide on new job assignments for the Test Robot. The digital computer is ideal for making logical analysis of performance results and varying job assignments by systematic, probabilistic, or combined rules. Some analog parts of the Simulation Robot Brain are already digitally controlled and, therefore, are available for job assignment tasks originated in the digital part.

Random Factors in Evaluation. In all systems, there are certain random factors. Analog simulations ordinarily use noise generators to introduce these. Since the noise generators will not repeat, it becomes difficult to determine whether the analog can give repeated solutions. Repeat solutions are often done to check on accuracy of the answers. The engineer may also want to repeat a particular job which resulted in failure. Sometimes it has been necessary to use magnetic tape records of noise to overcome this difficulty. In a digital computer, random factors are generated by a digital program called a pseudo-random number generator. This can be easily repeated. In these cases, it seems that the digital computer is the proper place to initiate random occurrences.

Philosophy of Equations

Analog specialists often consider continuous solutions to differential equations to be correct, and digital solutions to difference equations to be approximate. This is the result of the fact that, historically, calculus came before fast digital computers and physicists described the physical world by continuous models. Even so, they have been forced to use quantum, or discrete models in some areas. The test of a mathematical model of the physical world is to compare calculations with measurements of the physical world. It is quite possible that difference equations will give as good agreement as differential equations. Then the continuous solution to differential equations would justly be called an approximation.

Translators

An engineer who has been thoroughly bitten by the computer bug will want the Simulation Robot to understand plain English. In computer jargon, this means that the Simulation Robot must include a language translator to go from user-oriented language to machine-oriented language. The engineer would like to strap his creation to the flight table and tell the Simulation Robot, "This is supposed to work in X environment and accomplish Y. If it does not work, tell me why not." In order to translate this, the Simulation Robot must have self-consciousness, that is, it must know the types of things it can do. Here arises the first problem of translation; the Simulation Robot must be told by someone what it is and how to use the information. The Simulation Robot at NOTS is one of a kind and the same is true for the others presently in operation. No general-purpose ones seem to be envisioned for the near future. This means that a translator will have to be special for each Simulation Robot. An excessive amount of labor would be required for one to be written single-handed; it is to be hoped that a user's group will eventually be formed. In the meantime, existing translators can be used for the digital equipment and the use of digital programs to help in the setup of analog computers can be watched with interest. Then, ingenuity will be needed in using these aids to put together a specified environment for the Test Robot and in interpreting what happens, with perhaps help from the digital computer in tabulating results. In the meantime, it may be possible to plan a translator with general capability.

THE AUTOMATIC DETERMINATION OF HUMAN AND OTHER SYSTEM PARAMETERS

by T. F. Potts, G. N. Ornstein, and A. B. Clymer
North American Aviation, Inc., Columbus, Ohio

Summary

Automatic computer methods for the determination of system parameters from dynamic test data e.g., those of Meissinger, Leondes, Margolis, Graupe, Turner, and the authors, are reviewed and compared at length.

Two applications of a method of steepest descent (on the absolute magnitude of the system-equation-satisfaction error) are reported. In the first, simultaneous automatic analog determination of four coeffcients in a human response equation (transfer function) was accomplished. The second application of the technique was in an analog determination of aerodynamic coefficients from simulated flight test data.

The paper is concluded by a brief discussion of a broad range of potential applications of methods for determination of the parameter values required for computer simulation of human systems, other biological systems, socio-economic systems, and physical systems of concern in science and engineering.

Introduction

The class of problem addressed in this paper is that of determining the parameters of any given dynamic system by automatic methods on an analog or digital computer. In Section I, this class of problem is defined and discussed in relation to certain other important problems. In Section II many methods of solution are described and compared. Section III presents the method of steepest descent on the absolute magnitude of the error in satisfying the system equation. Two specific applications are presented in Sections IV and V. The paper is concluded in Section VI with a brief look at the benefits to science and technology which should result from further applications.

I. The Problem Of Parameter Determination

A simple instance of the parameter determination problem is that of finding some of the c_i coefficients in a linear equation

$$\sum_i c_i x_i = 0 \qquad (1)$$

given at least a sufficient amount of experimental data concerning the variables x_i and given the other coefficients c_i. One of the variables might be a forcing function. Other variables might be successive derivatives of a dependent variable, or values of a dependent variable at successively delayed times. However, the symmetry in equation (1) admits a dual interpretation, so that the forcing function might be among the unknowns.

The foregoing problem is readily generalized. For example, the system equation need not be linear; the system might require several equations for its description; the equations might contain unknown exponents and other constants as well as coefficients; and, moreover, the unknown parameters might not be constants. As will be discussed, however, nearly all of the methods which can solve the problem of equation (1) are theoretically capable of solving also these more general cases.

The problem of system parameter determination may be contrasted with the solution of a given differential equation, in which the coefficients and forcing function are known and in which the desired answer is the dependent variable as a function of the independent variable. In the parameter determination problem, on the other hand, one is usually given the system inputs and outputs and it is the set of system parameters which is desired.

Thus the problem of parameter determination is closely related to the electrical problem of synthesis, given the inputs and outputs of a black box. Indeed, the problem of parameter determination is a special case of the problem of system equation synthesis, (mathematical model optimization) in which not even the form of the system equation is originally given but in which it should be possible at least to ascertain some of the parameters (particularly exponents) by dimensional analysis, etc. Herein, however, it will be supposed that there is only a single system equation, that a reasonably valid form for the system equation is known, and that time histories of the variables are given, it being required to determine only the system equation parameters. This situation arises often in science, engineering, operations research, etc.

Parameter determination is allied also to a number of other important general problems in information processing, such as signal filtering, detection, recognition, and prediction. Indeed, the parameter-determination-problem concept may be a key to solution of these practical problems in many specific instances.

Another class of problems related to parameter determination is that of the calculus of variations. In the calculus of variations the unknowns are functions and/or end conditions. If the unknown functions are regarded as being characterized by a set of parameters, and if the unknown end conditions can be expressed in terms of parameters, then a problem in the calculus of variations can be recast as a problem in parameter determination. This possibility is especially attractive if the unknown function is required to be nonanalytic (eg, comprised of straight segments, in some solution space, representing delays, sudden changes, limitations, and other irregularities not easily treated by the classical calculus of variations).

It will be readily seen that the common problem of "function separation" falls under the class of parameter determination problems. There arises often a need for analysis of a given function into a sum of terms of known type but with unknown coefficients, exponents, etc. Examples of this function separation problem are radiochemical analysis of a mixture by identification of the half-lives of the constituents, determination of chemical kinetic functions within measured time histories of concentrations, separation of the normal modes in a dynamic structural response, etc.

The classical approach to the determination of system parameters was to devise a set of special experimental conditions which would isolate each parameter for direct measurement, all others being in terms contrived to be zero. Since this is not always feasible, the approach has been extended in many instances to the simultaneous determination of two parameters by indirect measurement, such as the measurement of natural frequency and damping ratio of a second order system in order to determine two of the three coefficients in the system equation.

Methods which determine at most one or two parameter values per experiment are excessively expensive and time consuming if the cost per test is high. Moreover, these methods are subject to error due to imperfect nulling of unwanted terms. A further difficulty is that closed-form solutions are not always available for use in the indirect methods of determining coefficients in a differential equation.

It is evident, then, that there has been need for methods which would permit simultaneous determination of many or all parameters of a system from a single dynamic experiment. Many such methods have been developed in the past few years and will be described below. These methods offer an optimization of testing and data reduction for minimum total cost.

II. Methods For Parameter Determination

Explicit Methods

The most readily understandable method of parameter determination is that of writing the system equation with variables evaluated at as many times as there are unknown parameters and then solving the resulting equations as a simultaneous set for the parameters. In the case of equation (1), for example, the simultaneous equations to be solved are:

$$\sum_i c_i x_i(t_j) = 0 \quad ; \quad j = 1, 2, \ldots \qquad (2)$$

Where the t_j are times at which data for the x_i are given, and where some of the c_i are the unknown parameters.

If the unknown c_i are functions of time, the process can be repeated with staggered sets of equations. This process lends itself well to a digital computer, and it is especially appropriate if the given data are sampled rather than continuous functions of time. The sampling could be either periodic or in accordance with some criterion relating to the derivatives of the variables or to the highest significant frequency in the spectrum of the forcing function. If the sampling interval is too short, the matrix is ill-conditioned, thus rendering the answers subject to large error due to amplified roundoff. On the other hand, if the sampling interval is too long, the higher frequency variations in the desired parameters will not be resolved. An unpublished preliminary investigation of this method on an operational analog computer was conducted in July 1960, by W. C. Franke, North American Aviation, Inc., Columbus Division, using circuits controlled by a clock circuit to yield solutions of 3 x 3 matrices staggered in time.

One of the shortcomings of the foregoing sampled matrix method is that it utilizes only sampled data. This is wasteful of information if the variables are known as continuous functions of time, as is the case in most dynamic experiments. Moreover, the sampled data used will yield erroneous results if an appreciable amount of noise is present in the measurements, since every datum is in an essential role. Accordingly, it would be desirable to use a method which employs a redundant amount of data, such as would be in short-term running averages instead of instantaneous data.

Redundant data can be made to yield exactly as many equations as there are unknowns even in methods of running regression analysis, such as "dynamic least squares". This method was investigated by one of the authors[1] in 1957-58 and has been applied widely[2]. In the application of the method of dynamic least squares it is necessary to choose the sampling interval and the number of samples in such a way that errors of measurement are smoothed out without causing loss of the higher frequencies in the desired parameters. One of the difficulties in the method of dynamic least squares is that the normal equations, being often nonlinear, may be troublesome to solve quickly and hence economically.

Another method for obtaining as many equations as there are unknown parameters is to differentiate analytically the system equation successively as many times as necessary. This possibility is not known to have been investigated, possibly because it would often require differentiation of noisy experimental data.

The use of as many sets of narrow-band filters as there are unknown parameters is another conceivable method using simultaneous equations. This method would fail in the case of a highly nonlinear system, since each discrete input frequency could excite a wide spectrum of discrete output frequencies.

It is often possible to use filters alone for parameter determination. An example is the case of the separation of dynamic structural response at a point into damped sinusoidal functions of time whose frequencies and damping ratios can be measured directly[3].

Another explicit method is cross-correlation, which is especially useful[4] when the system is represented in terms of a set of weights on successively delayed values of the forcing function[5,6] instead of in terms of an equation.

The classical statistical methods of multivariate regression analysis, factor analysis, multiple correlation analysis, etc., can be used for parameter determination in the case of systems which can be described by algebraic equations. Statistical methods developed by Albert[7], Wilkie [8,9], and others, have application in parameter determination and related problems.

Linear systems lend themselves to a variety of special methods. One which is in very common use in the process and control industries is the calculation of system equation coefficients from the break points and slopes of a Bode plot (log of amplitude response vs log of frequency of excitation). This method is suited to automation on a digital computer. Such automation is often facilitated by the fact that a suitable system model can be of a much lower order than the system itself (at least under certain conditions[10].

Implicit Methods

The foregoing methods yield a direct solution for the unknown parameters. In contrast, the methods to be discussed next involve an iterative or continuous approach to the desired parameters.

As long as there have been analog computers they have been used in a trial-and-error process to match experimental results and thereby to determine system parameters. The published applications which have been made of this manual tentation method are far too numerous to list. It is laborious, and it is subject to errors of human judgment. Therefore an automatic and objective method would be much to be preferred.

A simple and more objective method is to let the computer determine the mean-squared error of fit of the computer-generated dependent variable and the given time history of the dependent variable. By judicious adjustment of the unknown parameters it is possible to minimize the mean-squared error. This process can be automated by closing the loop on the computer, a method which has been widely used[11]. As automated, however, it can determine only one parameter per equation (i.e., per dependent variable) per run.

A related method is "implicit synthesis"[1], in which a desired parameter is adjusted automatically to minimize the instantaneous error in the dependent variable. Among the interesting applications of implicit synthesis which have been made are the determination of lift coefficient during transient stall, aircraft-tire friction during landing, and nuclear reactor parameters. Implicit synthesis yields, in general, only one parameter per equation. However, W. C. Franke (in an unpublished study) has shown that under certain conditions it is possible to obtain a second parameter simultaneously by letting it be adjusted automatically to minimize the instantaneous error in the rate of change of the dependent variable.

The first published implicit nonrandom iterative method known to the authors which is capable of yielding several parameters at once per equation is one due to Meissinger[12]. It is basically steepest descent on an integral of the square of the error in the dependent variable. It has the disadvantage that it requires the use of auxiliary differential equations. It has been applied, nevertheless, to a number of problems including adaptive servos[13,14].

One of the characteristics of all of the foregoing implicit methods of parameter determination is that they contain a mathematical model of the system, which has been called a "learning model"[13] since it "learns" the correct parameter values. The learning model generates continually a tentative value of the dependent variable for comparison with experimental data.

Automatic determination of parameters can be accomplished also by steepest descent on some even function of the error in satisfaction of the system equation. This class of methods has been investigated and applied by Graupe[15] and independently by the authors in unpublished work during 1960. Some of the earlier specific applications of methods in this class have been discussed by Turner[16] and Levine[17]. It is this class of methods with which Sections III-V are concerned.

The authors' studies of steepest descent on equation error began in March 1960 with the concept that an excessively large value for a coefficient in the learning model would cause the "signature" of the faulty term to show up in the equation error. Then one would expect the quantity $\frac{1}{t}\int_0^t \varepsilon \cdot x_i \, dt$ to have a positive value, where ε is the error in the system equation. A term in the system equation having a correct value for its coefficient would give a small or zero value for the above quantity. Therefore this quantity could be used as a measure of the rate at which the coefficient of

x_i should be reduced between runs in an iterative procedure. It was recognized that a continuous correction could be obtained by using a running integral over a finite period or by using the output of a first order lag circuit instead of the above quantity. The latter possibility was investigated on an analog computer in June 1960 in the case of a third order system equation and was found to be successful in driving out good values of all four coefficients in five seconds. It was found, moreover, that the lag circuit had little effect, equally satisfactory performance being obtained without it. Without the lag circuit this method is simply steepest descent on the square of the equation error, as was recognized after the authors had seen Meissinger's paper[12].

Noting, in continuous steepest descent on the square of the equation error, that the rate of convergence to the answer became progressively slower in the manner of a decaying exponential, the authors tried the scheme of sampling and holding new values of the variables whenever the equation error had been reduced to a prescribed level. This gave improved convergence.

Next it was noted that the multipliers were acting essentially only as sign changers, so they were replaced with relay-amplifier sign changers, which converted the method to steepest descent on the absolute magnitude of the equation error. At about this time (September 1960) Graupe's preprint[15] was sent to the authors for discussion.

Independently, Graupe starting with implicit synthesis, had been investigating several of the same methods and applying them to nonlinear problems. His example problems were solved by his methods with time constants ranging from 6 secs. down to as little as 0.1 sec. He developed also more general methods of steep descent, one of which subsumes all methods of continuous steep descent on an even function of the equation error. Graupe studied also the case of a system described by more than one system equation. One of Graupe's more interesting remarks is that steepest descent can be used for the determination of the highest derivative in a differential equation in which the highest derivative cannot conveniently be isolated analytically for explicit calculation and successive integration.

It is instructive to note relationships among the methods discussed above and with certain other methods. For example, it has been shown in unpublished analyses by B. J. Miller, North American Aviation, Inc., Columbus Division, that an analog circuit for implicit synthesis is in some cases identical with that for steepest descent on a function of the system equation error.

Linear programming is closely related to parameter determination. As performed on an operational analog computer[18], linear programming is simply steepest descent (or ascent) on an "objective function" of the parameters c_i, with the complication that there are in general some constraints on the parameters. If the objective function is nonlinear, the same approach is applicable. Even time-varying constraints can be handled, thus permitting solution of a class of problems in dynamic programming. Thinking of the solution of a linear programming problem as being the optimum set of values of the parameters of an "operating system" (as distinguished from a "hardware system"), one can then conceive by analogy that steepest descent can be applied to the optimization of the parameters of a hardware system. This possibility has been pointed out by Meissinger[12] and has been applied by Margolis and Leondes[13,14], thus establishing a close relationship between parameter determination methods and the "optimizers" which are being developed for process control systems[19-22].

There is no particular virtue in steepest descent as such. What is really desired is to move as rapidly as possible in error-parameter space in the direction of the answer rather than merely in the steepest direction. It is interesting to note that the long-term average of the instantaneous steepest directions is the direction to the answer. Accordingly, it might be advantageous to employ incremental descents in averaged directions, or continuous descent in the running average direction, rather than steepest descent. In the latter case, if the running average is weighted in an exponentially decaying manner with time measured backwards from the present, one has the method mentioned earlier as involving a first order lag circuit. These possibilities have not been investigated by the authors.

The following section presents a general exposition of the method of continuous steepest descent on the absolute magnitude of the error in satisfying the system equation. This method is that utilized in the application of Sections IV and V.

III. A Parameter Determining Technique

Consider the linear differential equation

$$a_0 x + a_1 \dot{x} + \cdots + a_m \overset{m}{x} = F(t) + b_1 \dot{F}(t) + \cdots + b_n \overset{n}{F}(t) \qquad (3)$$

where the coefficients are not necessarily constants, but may be functions of time and/or of x, F(t) or their derivatives. A method for determining the coefficients a_i, b_j is sought--a method requiring only that F(t) and $x(t)$ be given.

First, let a_{ci} and b_{cj} represent the computed values of coefficients a_i and b_j, respectively, at any time t. Then, define the equation error

$$\varepsilon = \sum_{i=0}^{m} a_{ci} \frac{d^i x}{dt^i} - \sum_{j=1}^{n} b_{cj} \frac{d^j F(t)}{dt^j} \qquad (4)$$

Now at any time t, the specified values of X, $\dot{X}$, $\ddot{X}$, ..., $\overset{m}{X}$, F(t), $\dot{F}(t)$, $\ddot{F}(t)$, ..., $\overset{n}{F}(t)$ define an m+n+2 dimensional hypersurface given by

$$|\varepsilon| = \left| \sum_{i=0}^{m} a_{ci} \frac{d^i x}{dt^i} - \sum_{j=1}^{n} b_{cj} \frac{d^j F(t)}{dt^j} \right| \qquad (5)$$

where $\mathcal{E}$ and the a_{ci} and b_{cj} are the coordinates of a running point in the m+n+2 dimensional error-parameter space. Further, the hyperplane $\mathcal{E}=0$ (of dimension m+n+1) is tangent to this hypersurface along some hyperplane (of dimension m+n) within $\mathcal{E}=0$. Thus the point of intersection of all tangency locus hyperplanes of dimension m+n in $\mathcal{E}=0$ (as the hypersurface moves and deforms in time) is sought. The coordinates of that point will be (a_0, a_1, ...a_m, b_1, ... b_n, and $\mathcal{E}$ (=0)).

We now proceed to find the required point of intersection. Form the slopes in the error-parameter hyperspace as follows:

$$\left.\begin{aligned} \frac{\partial|\mathcal{E}|}{\partial a_{c0}} &= \frac{\mathcal{E}}{|\mathcal{E}|}\cdot X \\ \vdots \quad & \quad \vdots \\ \frac{\partial|\mathcal{E}|}{\partial a_{cm}} &= \frac{\mathcal{E}}{|\mathcal{E}|}\cdot \overset{m}{X} \end{aligned}\right\} \qquad (6)$$

$$\left.\begin{aligned} \frac{\partial|\mathcal{E}|}{\partial b_{c1}} &= \frac{\mathcal{E}}{|\mathcal{E}|}\cdot \dot{F}(t) \\ \vdots \quad & \quad \vdots \\ \frac{\partial|\mathcal{E}|}{\partial b_{cn}} &= \frac{\mathcal{E}}{|\mathcal{E}|}\cdot \overset{n}{F}(t) \end{aligned}\right\} \qquad (7)$$

Thus one may, considering these slopes, drive the running point "downhill" along the path of steepest descent toward the nearest set of values of the parameters for which the absolute value of the error is minimum (zero). Specifically, if the projection of the running point into the $\mathcal{E}=0$ hyperplane is made to move along the normal to the tangency locus hyperplane (of dimension m+n) and toward the latter hyperplane, a uniform convergence toward (a_0, a_1, ..., a_m, b_1, b_2, ..., b_n) is assured, since when the running point has reached the tangency locus hyperplane it is closer to the point of intersection than it was when it started. Thus rates of correction to the parameters may be specified for descent in the steepest direction by choosing them to be proportional to the slopes of the hypersurface in the coordinate directions:

$$\left.\begin{aligned} \dot{a}_{c0} &= -G\cdot\frac{\mathcal{E}}{|\mathcal{E}|}\cdot X \\ \vdots \quad & \quad \vdots \\ \dot{a}_{cm} &= -G\cdot\frac{\mathcal{E}}{|\mathcal{E}|}\cdot \overset{m}{X} \\ \dot{b}_{c1} &= -G\cdot\frac{\mathcal{E}}{|\mathcal{E}|}\cdot \dot{F}(t) \\ \vdots \quad & \quad \vdots \\ \dot{b}_{cn} &= -G\cdot\frac{\mathcal{E}}{|\mathcal{E}|}\cdot \overset{n}{F}(t) \end{aligned}\right\} \qquad (8)$$

where G is a large number (gain).

Example

Consider the equation

$$a_0 X + a_1 \dot{X} = F(t) \qquad (9)$$

We seek to determine a_0 and a_1 where F(t), x(t), and $\dot{x}$(t) are known.

First, define the system equation satisfaction error,

$$\mathcal{E} = a_{c0} X + a_{c1}\dot{X} - F(t) \qquad (10)$$

where a_{c0} and a_{c1} are the computed values of the coefficients. Now geometrically (see Figure 1)

$$|\mathcal{E}| = | a_{c0} X + a_{c1}\dot{X} - F(t) | \qquad (11)$$

represents a surface opening upward from the a_0a_1 plane. This surface is tangent to the plane in the line A given by

$$a_{c0} X + a_{c1}\dot{X} - F(t) = 0 \qquad (12)$$

whence the slopes of the surface along the a_{c0} and a_{c1} axes (i.e., the components of the gradient vector) are

$$\frac{\partial|\mathcal{E}|}{\partial a_{c0}} = \frac{\mathcal{E}}{|\mathcal{E}|}\cdot X \qquad (13)$$

$$\frac{\partial|\mathcal{E}|}{\partial a_{c1}} = \frac{\mathcal{E}}{|\mathcal{E}|}\cdot \dot{X} \qquad (14)$$

If the coordinates of the running point are driven at rates proportional to the negatives of these slopes, the running point moves toward the line A along a path normal to that line. As the values of X, $\dot{X}$, and F change with time the tangent line will rotate about some fixed point having coordinates (a_0, a_1). The running point, however, will converge uniformly upon the (a_0, a_1) position as noted above.

IV. Application Of The Method To Determination Of Human Coefficients

Background

One area of application of the above described technique arises in the determination of an equation representing the human operator as a control element[23]. This area has been investigated in the past by utilization of classical engineering (control system) techniques. The techniques used range from analytic transfer function and cross-spectrum approaches (24-28) to the simulation approach involving the use of electronic computers (29-31). A recent report by McRuer and Krendel[32] presents an excellent review of studies in which these various techniques have been applied. All of these techniques seek to express the output of the human operator as a function of input to him.

One factor which has become quite apparent is that none of these techniques permits the rapid and efficient determination of parametric values for a human transfer function. For example, the describing function and quasi-linear models require at least a cross-spectrum analysis and computerized mathematical/statistical programs. Even the simulation technique as developed at the Goodyear Aircraft Co.[29,30] involves a process of successive iteration in order to match simulated human output to actual human output. The only application of the simulation approach in which an attempt was made to estimate automatically the human transfer function parametric values was in the work of Fuchs.[31] Fuchs was successful in automatically estimating, one at a time, the coefficients of a second order linear differential equation representing part of a human transfer function. All remaining values of the coefficients were required to be pre-specified.

The application to be discussed involves the simultaneous determination of four coefficients in a human transfer function. The technique described in Section III was employed in two experiments involving the human operator as a controller in a complex servo loop. Specifically, variations in the values of the human transfer function parameters were determined as a function of known sets of system characteristics. The first experiment treated the influence of display quickening upon the parameters of the human response equation, while the second experiment treated the influence of damping upon these parameters.

The equation form chosen for the human transfer function was

$$HTF = \frac{1 + a s}{b + c s + d s^2} e^{-.2s} \qquad (15)$$

where a, b, c, and d were the coefficients to be determined. This is the linear component of the quasi-linear model exercised by McRuer and Krendel.

Apparatus

A block diagram of the entire system apparatus is presented in Figure 2. For descriptive purposes, three major elements may be recognized. First, there is the tracking device. Involved here are display and control hardware and the various elements needed to produce an input signal, perform proper operations upon the control stick output, and present an error signal to the subject. Second, there is what shall be termed the synthesizing circuit, which performs the determination of the coefficients in the human response equation. Third, there is a circuit which comprises the simulation of a human transfer function which was used in evaluating the synthesizing circuit.

Synthesizing Circuit

Figure 3 presents a block diagram of an analog circuit designed to implement the human coefficient determining technique. In the described circuit there exists at any time t a set of values of $\theta_o(t)$, $\dot{\theta}_o(t)$, $\ddot{\theta}_o(t)$, $e(t-.2)$, $\dot{e}(t-.2)$ and $\mathcal{E}(t)$. Based upon these values a set of coefficients (a_c, b_c, c_c and d_c) is computed which puts the running point on the tangent hyperplane, thus driving $\mathcal{E}$ toward zero.

The system equation satisfaction error, is defined by

$$\mathcal{E} = b_c\,\theta_o(t) + c_c\,\dot{\theta}_o(t) + d_c\,\ddot{\theta}_o(t) - e(t-.2) - a_c\,\dot{e}(t-.2) \qquad (16)$$

The 0.2 second delay was obtained from a second order Padé approximation. It will be noted that with the circuit as described, the coefficients obtained will contain contributions from the control element. These can easily be removed if the control element transfer function is known.

Reliability and Validity of Coefficient Determination

The reliability and validity of the determination of coefficients in an equation of the form of equation (15) was investigated. An analog circuit representing equation (15) and having controllable coefficients was placed in the overall network in a manner such that it could be substituted (see Figure (2)) for the human operator. A series of runs was then made in which known (human-like) values were set into the "manalog", a circuit representing equation (15); and during these runs the synthesizing circuit determined the values of coefficients in its usual manner. Inasmuch as the circuit was found in pilot studies to require approximately 30 sec. to converge to an asymptotic value for a human subject, runs were of 1.5 min. duration, the values of the coefficients being averaged over the last minute. Table I summarizes the results of this series of runs. It may be seen that the coefficient-determining circuitry did not produce precisely the desired values of the coefficients--even though the true variability of any given coefficient was nil. The observed variability over runs was attributed to the fact that the manalog, which exhibits smoother performance than the human, did not uniformly reach final coefficient values during the one and one-half minute trial due to the fact that convergence rate is a function of the magnitude of the error signal and its derivatives which are randomly different over trials.

Based upon this set of observations, an iterative procedure designed to circumvent the convergence problem was devised, as follows: after each trial the coefficients therein determined were set as initial conditions in the coefficient-determining circuits. This procedure was used effectively throughout all runs of the experiment. Table II presents the data for five sequential manalog runs using the iterative process. The coefficients are stable to within .001 after two iterations.

The forcing function $\theta_i(t)$ was randomly generated, which resulted in a forcing function rms which varied greatly from trial to trial. In order to standardize the forcing function input over subjects it was decided to average over four trials for each subject under each condition. This procedure resulted in an rms which, based upon a sample of 100 trials, varied by a maximum factor of 1.3 over 25 blocks of four trials. This factor represents the ratio of the highest rms (based upon four randomly selected trials) to the lowest similarly based rms. The value 1.3 may be compared with a single trial maximum ratio value of 2.1.

The reliability of the technique under the averaging procedure was determined by estimating the variance of each of the coefficients. Forty manalog trials were run with initial conditions in the coefficient determining circuit set at the true values indicated in Table II. These trials were then averaged in groups of four. The resulting ten sets of coefficients yielded standard deviations as follows:

$$\sigma_a = .0012; \sigma_b = .0005; \sigma_c = .0007; \sigma_d = .0004$$

The same data as described in the previous paragraph permit a comparison of average determined value and true value. Coefficients a, b, c, and d when averaged over the forty trials were found to have values: .2008, .0507, .2504 and .003 respectively.

General Experiment Design

Serving as subjects were four male volunteers; two jet flight test pilots, one USAF-reserve jet pilot, and one non-pilot. Subjects were run individually in the tracking device previously described.

Experimental trials were administered in blocks of five with about 90 sec. rest between trials. During the 90 sec. interval the various metrics were recorded, and the iterative initial conditions procedure discussed above was implemented. After each block of trials a rest of from 5 - 10 min. occurred.

Experiment I

One of the major efforts to consider the human as a functional servo system is the work of Taylor et al. at the Naval Research Laboratory[33-36]. Perhaps the best statement of this approach appears in the now-classic paper by Birmingham and Taylor[33] wherein the approach is set forth as what they term a basic principle of control design. In essence, this principle states that, when designing man-machine control systems, man should be required to act no more complexly than does a simple amplifier.

One application of the Birmingham and Taylor principle results in a system that is said to be quickened. Figure 4 presents two systems -- one quickened, the other not quickened. Specifically, system A is not quickened; the human views an error signal which is e = $\theta_i - \theta_o$. In system B, the quickened system, the human views an error signal e = $\theta_i - [\theta_o + K_1\dot{\theta}_o + K_2\ddot{\theta}_o]$ which is seen to contain derivative information. Thus the subject, when using a quickened display, is provided directly with derivative information.

Now, according to the Birmingham and Taylor hypothesis, the quickened display should result in "better" performance because the human, when using such a display, is required to do less differentiating (rate estimation)--derivative information being already inherent in the displayed signal, e. If this were the case, the prediction would follow that the human response equation derived under conditions involving the use of a quickened display should show a weighting of the $\dot{e}$ term decreased from that obtained under conditions wherein an unquickened display was used. Further, it might also be predicted that the condition of Partial Quickening ($K_2 = 0, K_1 > 0$ in Figure 4) should yield a weighting of $\dot{e}$ intermediate to those obtained under the No Quickening ($K_1 = 0$, $K_2 = 0$) and Full Quickening ($K_1 > 0$, $K_2 > 0$) conditions. An experiment was performed to test these predictions.

The general details of method were those discussed above. The non-zero values of the quickening coefficients were taken as $K_1 = .5$ and $K_2 = .25$. No spring was attached to the control stick nor was the damper attached. Conditions were presented to subjects in a randomized order with the restriction that the No Quickening condition appeared first.

The weights assigned to the $\dot{e}$ term of the human response equation under the three quickening (averaged over subjects) conditions are summarized in Figure 5. The differences between the No Quickening and Partial Quickening conditions, and between the No Quickening and Full Quickening conditions are significant beyond the .01 level (t = 7.31 and 7.02, respectively, with 3 df). A similar examination of the other coefficients of the human response equation (see Equation 15) failed to reveal other differences of significance at this level.

Experiment II

Whereas Experiment I treated display factors, this experiment is concerned with the effects upon the human response equation of variations in the amount of control stick damping. The rationale, however, derives as in Experiment I from the work of Birmingham and Taylor. Consider a system as in Figure 4A, each element having a given response equation. Suppose now that an additional integrator is placed in the system and located in the box labeled "control stick". If the system without this addition tracks asymptotically at a given level of accuracy then the insertion of the integrator as described will require some change in one of the other system elements if the same level of tracking accuracy is to be maintained. Inasmuch as a damper on the control stick performs an integration, an experiment suggests itself wherein the mechanism remains unchanged and the control stick is changed by varying its coefficient of damping. Under these conditions the human is predicted to change his response equation so as to compensate for the inserted integration. Further, it is postulated that the amount of change in the human response equation will be related to the amount of damping inserted. Thus, three conditions are considered: No Damping, Light Damping, and Heavy Damping.

As in Experiment I the general details of method were those discussed above. A nonquickened display was utilized and both a spring and damper inserted. The spring constant was 1.2 lb./deg.

and the time constant of the damper was adjusted to the values 0.000, 0.05, and 0.28 sec. for the conditions No Damping, Light Damping, and Heavy Damping, respectively. The same subjects were utilized as in Experiment I, which was concluded before the start of the present experiment. The order of conditions was randomized over subjects.

Figure 6 presents the weights assigned to the ė term of the human response equation under the three conditions of damping. All intercondition differences are significant at beyond the .01 level (t = 40.6, 16.8, 29.2 for No Damping vs. Heavy Damping, No Damping vs. Light Damping, and Light Damping vs. Heavy Damping, respectively, with 3 df). No comparisons across conditions were significant for the other coefficients of the human response equation.

Summary of Application I

This application demonstrated the utility of a technique for the automatic analog determination of human response equation parameters. The technique has proven efficient and reliable, and provides meaningful metrics of performance.

V. Application Of The Method To Determination Of Aerodynamic Coefficients

Background

Another promising application of the technique is in the area of automatic reduction of flight test data to aerodynamic coefficients. A great deal of effort has been expended on this problem in recent years, such as references 8 and 9, but there is still much room for improvement in the methods available for use. It is to be hoped that the application of the method described herein will bring an improvement to the state-of-the-art.

The tests were performed using simulated flight test data. The two degree-of-freedom constant-speed lift and pitch aircraft equations of motion were solved on the analog computer to provide simulated flight test time histories of pitch rate, angle of attack and their first derivatives. These signals were operated on according to the above described technique to yield the aircraft dimensional stability coefficients. To obtain the usual non-dimensional aerodynamic coefficients only algebraic operations on the dimensional coefficients are required.

Description of Apparatus and Method

A block diagram of the apparatus used is presented in Figure 7. The two equations representing the aircraft simulation are:

$$\ddot{\theta} + M_{\dot{\theta}}\dot{\theta} + M_{\alpha}\cdot\alpha + M_{\dot{\alpha}}\cdot\dot{\alpha} = M_{\delta h}\cdot\delta h \qquad (17)$$

$$\dot{\theta} - L_{\alpha}\cdot\alpha - \dot{\alpha} = L_{\delta h}\cdot\delta h \qquad (18)$$

$M_{\dot{\theta}}$, M_{α}, $M_{\dot{\alpha}}$, $M_{\delta h}$, L_{α}, and $L_{\delta h}$ represent the unknown coefficients to be determined. In this case, the form of the equations describing the system is known explicitly. Two system equation satisfaction errors can be formed,

$$\varepsilon_1 = \ddot{\theta} + M'_{\dot{\theta}}\cdot\dot{\theta} + M'_{\alpha}\cdot\alpha + M'_{\dot{\alpha}}\cdot\dot{\alpha} - M'_{\delta h}\cdot\delta h \qquad (19)$$

$$\varepsilon_2 = \dot{\theta} - L'_{\alpha}\cdot\alpha - \dot{\alpha} - L'_{\delta h}\cdot\delta h \qquad (20)$$

where the primes denote the computed values of the coefficients. Considering $|\varepsilon_1|$ and $|\varepsilon_2|$ to represent hypersurfaces, as in the previous discussion, the components of the gradients can be written as

$$\left.\begin{aligned} \frac{\partial|\varepsilon_1|}{\partial M'_{\dot{\theta}}} &= \frac{\varepsilon_1}{|\varepsilon_1|}\cdot\dot{\theta} \\ \frac{\partial|\varepsilon_1|}{\partial M'_{\alpha}} &= \frac{\varepsilon_1}{|\varepsilon_1|}\cdot\alpha \\ \frac{\partial|\varepsilon_1|}{\partial M'_{\dot{\alpha}}} &= \frac{\varepsilon_1}{|\varepsilon_1|}\cdot\dot{\alpha} \\ \frac{\partial|\varepsilon_1|}{\partial M'_{\delta h}} &= \frac{-\varepsilon_1}{|\varepsilon_1|}\cdot\delta h \end{aligned}\right\}(21) \qquad \left.\begin{aligned} \frac{\partial|\varepsilon_2|}{\partial L'_{\alpha}} &= \frac{-\varepsilon_2}{|\varepsilon_2|}\cdot\alpha \\ \frac{\partial|\varepsilon_2|}{\partial L'_{\delta h}} &= \frac{-\varepsilon_2}{|\varepsilon_2|}\cdot\delta h \end{aligned}\right\}(22)$$

Thus, to drive the computed coefficients against the gradient, toward values yielding zero system equation satisfaction errors, the rates of change of the coefficients are taken to be

$$\left.\begin{aligned} \dot{M}'_{\dot{\theta}} &= -G\cdot\frac{\varepsilon_1}{|\varepsilon_1|}\cdot\dot{\theta} \\ \dot{M}'_{\alpha} &= -G\cdot\frac{\varepsilon_1}{|\varepsilon_1|}\cdot\alpha \\ \dot{M}'_{\dot{\alpha}} &= -G\cdot\frac{\varepsilon_1}{|\varepsilon_1|}\cdot\dot{\alpha} \\ \dot{M}'_{\delta h} &= +G\cdot\frac{\varepsilon_1}{|\varepsilon_1|}\cdot\delta h \end{aligned}\right\}(23) \qquad \left.\begin{aligned} \dot{L}'_{\alpha} &= +G\cdot\frac{\varepsilon_2}{|\varepsilon_2|}\cdot\alpha \\ \dot{L}'_{\delta h} &= +G\cdot\frac{\varepsilon_2}{|\varepsilon_2|}\cdot\delta h \end{aligned}\right\}(24)$$

where G represents a fixed gain chosen experimentally for good convergence.

Studies Performed

The convergence of the coefficients was investigated under conditions wherein the coefficients were constants. Forcing function inputs to the horizontal stabilizer deflection, comprised sine, triangular and square waves of various frequencies and amplitudes, random noise of various power spectral densities, and combinations of random and periodic signals.

Results

Some general observations on the convergence of the computed coefficients when the true coefficients were all constant can be noted. The rate of convergence was found to be directly proportional to the amplitude of the forcing function and to the loop gain of the coefficient synthesizing loops. The noise level on the computed coefficients was also found to be proportional to these factors, but fortunately a good compromise could be reached here, such that an exponential convergence time constant of one second could be obtained with reasonable noise levels on the coefficients and with realistic horizontal stabilizer inputs. The ability of the systems to converge uniformly to the correct values of the constants was found to be a function of the power spectral density of the forcing function; in general, it was found that it was necessary to offset large amounts of low frequency power with large amounts of high frequency power to prevent the computed coefficients from tracking the forcing function waveform. Using periodic forcing function inputs, it was found that only one narrow frequency range gave good convergence of the coefficients.

Using a forcing function input yielding an exponential convergence time constant of one second, the coefficients were allowed to converge to their correct values and then the M coefficient was instantaneously doubled. The corresponding computed coefficient assumed its new value in about one second, with only minor momentary disturbances to the other coefficients.

Summary of Application II

The results indicate that the technique shows promise for the automatic reduction of flight-test data to aerodynamic coefficient form.

VI. Other Possible Applications

Other Applications to Human Simulation

The determination of human "tracking-situation" coefficients discussed above is clearly only one of the many possible instances of the application of automatic methods for human system parameter determination. These methods comprise a convenient preliminary to human simulation on a computer. Similarly automatic determination of the parameters of more elaborate theoretical models of the human operator[37,38] would be desirable as a step toward development of control systems modeled after the human by first simulating the human operator in detail[39,40].

In the field of human physiology there are still many homeostatic (regulatory) systems for which differential equations and parameters of desired accuracy are lacking[37,41], in spite of the numerous quantitative experimental studies and simulations already performed[42,43]. Automatic parameter determination methods in conjunction with differential equations derived from theoretical models should greatly expedite the realization of human physiology simulations involving several homeostatic systems and their interactions.

One of the many possible practical applications of automatic methods to physiological parameter determination would be a device for use in operating rooms to serve as a monitor to warn the anesthetist and/or surgeon in the event of significant changes in critical parameter values of a patient. A similar device might find application to pilots of future flight vehicles.

Simulation of social, economic, ecological, industrial, demographic and political systems[37] is handicapped at present because of a lack of means for deriving the values of parameters from available data and because of the difficulty or impossibility of performing experiments designed to simplify parameter determination. Untold benefits await this vast field of applications of methods for automatic determination of parameters.

Insofar as higher processes in the human brain may be better described by some as-yet-undeveloped branch of logic than by mathematical equations the present techniques are not applicable. However, insofar as mathematical equations suffice, a new and powerful tool is available.

Other Applications In Physical Science and Technology

The determination of aerodynamic coefficients as discussed herein is merely illustrative of the possibilities for application of automatic methods for parameter determination in systems of concern in the physical sciences and in engineering. A lengthy list of some of these possibilities has been published previously.[1] One may in addition note the possibility of determining the parameters of:

1. Equations for boundary layers, wakes, noise fields, etc., either as kinematic field descriptions alone or with physical constraints as imposed, e.g., by Lagrange's equations.
2. Transient aerodynamic functions occurring during transient stall, V/STOL maneuvers, etc.
3. Theoretical models developed by operations research.
4. Control strategies and other high-level heuristics and routines in computers of the future[44].
5. Living systems studied for the purpose of designing analogous engineering devices[45].
6. Signal signature functions.
7. Molecular structure as revealed by infrared spectra.

VII. Conclusions

1. The methods discussed herein afford the means for a revolution in the science of dynamic measurement. Instead of it being necessary to conduct numerous specialized test runs on a system, it is now sufficient to run only

one test (or certainly fewer tests than formerly) in order to determine simultaneously all of the parameters of a system[1].

2. These methods should serve as a powerful stimulus to analog, digital, and hybrid computer simulation of systems of all kinds. The benefits of the computer simulations which will result may be readily appreciated.

3. These methods have been demonstrated sufficiently to justify the investment of research in further efforts to seek and develop improved methods for parameter determination, for the design of dynamic experiments, and for system equation determination in general.

Acknowledgments

This work has benefited greatly from the technical contributions of Messrs. W. C. Franke, C. R. Walli, B. J. Miller, and G. W. McClary. Their contributions are gratefully acknowledged.

References

1. Clymer, A.B., "Direct System Synthesis by Means of Computers", Communication and Electronics (AIEE Transactions), Jan. 1959, P. 798-806.

2. Guier, W.H., and Weiffenback, G.C., "The Doppler Determination of Orbits", NASA Conference on Orbit and Space Trajectory Determination, 12 March 1959.

3. Beals, V.L., and Hurley, S.R., "The Application of Impulsive Excitation to In-Flight Vibration Testing", IAS Paper 60-71. Presented at IAS National Summer Meeting, Los Angeles, June 28-July 1, 1960.

4. Miller, D.R., "Applications of Analog Computers to the Paper Industry", Central-Midwest Simulation Councils meeting, St. Louis, 3 February 1958.

5. Goodman, T.P., and Reswick, J.B., "Determination of System Characteristics from Normal Operating Records", Trans. ASME, vol. 78, 1956, pp 259-71.

6. Janssen, J.M.L., and Ensing, L., "The Electro-Analogue, An Apparatus for Studying Regulating Systems", Phillips Technical Review, March 1951 (reprinted in "A Palimpsest on the Electronic Analog Art", H. M. Paynter, Ed., Geo. A. Philbrick Researches, Inc., Boston, Mass, 1955).

7. Albert, G.E., "Statistical Methods in Prediction, Filtering, and Detection Problems" Journal of the Society for Industrial and Applied Mathematics, Volume 8, No. 4, Dec. 1960. pp. 640-653.

8. Wilkie, L.E., "Final Report on Extraction of Aircraft Stability Coefficients from Flight Test Data and Theoretical and Experimental Studies on Selected Problems of High Speed Aerodynamics and Dynamic Stability," USAF WADC TR 57-723, Dec. 1957.

9. Wilkie, L.E., et al, "The Determination of Aircraft Stability Coefficients from Flight Test Data", USAF WADC TN 57-410, Parts I, II, III, 1958.

10. Thal-Larsen, H., and Takahashi, Y., "Identification of Process Parameters by Means of Models", ASME Paper No. 60-WA-121, Dec. 1960.

11. Laning, J.H., Jr., and Battin, R.H., "Random Processes in Automatic Control", McGraw-Hill, New York, 1956, p. 253ff.

12. Meissinger, H.F., "The Use of Parameter Influence Coefficients in Computer Analysis of Dynamic Systems", Proceedings of the Western Joint Computer Conference, San Francisco, May 1960.

13. Margolis, M., and Leondes, C.T., "A Parameter Tracking Servo for Adaptive Control Systems" IRE Transactions on Automatic Control, Vol. AC-4, No. 2, Nov. 1959, p 100-111.

14. Margolis, M., and Leondes, C.T., "On the Theory of Adaptive Control Systems, the Learning Model Approach", Congress of the International Federation for Automatic Control, Moscow, June 1960.

15. Graupe, K.K., "The Analog Solution of Some Functional Analysis Problems", AIEE Paper No. 60-1230 (in press).

16. Turner, R.M., "On the Reduction of Error in Certain Analog Computer Calculations by the Use of Constraint Equations", Proceedings of The Western Joint Computer Conference, San Francisco, May 1960.

17. Levine, L., "Analog Setup...1. Solves Polynomials 2. Plots Root Locus...Automatically", Control Eng., Oct. 1960, pp. 125-6.

18. Pyne, I.B., "Linear Programming on an Electronic Analogue Computer", Communication and Electronics (AIEE Transactions), 1956, p. 139-143.

19. Andrew, A.M., "Learning Machines", in "Mechanisation of Thought Processes", National Physical Laboratory Symposium No. 10, Her Majesty's Stationery Office, 2 vols., 1959.

20. Feldbaum, A., "An Automatic Optimizer", Automatika i Telemekhanika, 19:731-43, Aug. 1958 (translated in Automation Express, Sept. 1958).

21. Junson, J. K., and Rubin, A.I., "Optimization by Random Search on the Analog Computer" IRE Transactions on Electronic Computers, Vol. E C--8, No. 2, June 1959, P. 200-203.

22. Wheeling, R.F., "Optimizers; Their Structure" Communications of the ACM, Dec. 1960, P. 632-638.

23. Ornstein, G.N., "Applications of a Technique for the Automatic Analog Determination of Human Response Equation Parameters", North American Aviation, Inc., Columbus Division. Report Nos. NA61H-1, January, 1961.

24. Tustin, A., "The Nature of the Operator's Response in Manual Control, and Its Implications for Controller Design", J. Inst. Elec. Engrs., London, 1947, 94, 190-207.

25. Ragazzini, J.R., "Engineering Aspect of the Human Being as a Servo-mechanism", paper read at Amer. Psychol. Assn. Meeting, Boston, Sept. 1948.

26. Russell, L., "Characteristics of the Human as a Linear Servo-element" Masters Thesis, MIT, 1951.

27. Benepe, O.J., Narosimhon, R., and Ellson, D.G., "An Experimental Evaluation of the Application of Harmonic Analysis to the Tracking Behaviour of the Human Operator", USAF WADC TR 53-384, May, 1954.

28. Elkind, J.I., "Characteristics of Simple Manual Control Systems" Lincoln Laboratory, MIT Technical Report III, April, 1956.

29. Goodyear Aircraft Co., "Final Report, Human Dynamics Study", Goodyear Aircraft Company, Akron, Ohio GER Report 4750, 1952.

30. Goodyear Aircraft Co., "Investigation of Control 'Feel' Effects on the Dynamics of a Piloted Aircraft System", Goodyear Aircraft Co., Akron, Ohio, GER Report 6726, 1955.

31. Fuchs, A.H., "The Progression-Regression Hypotheses in Perceptual - Motor Skill Learning", Doctor's dissertation, The Ohio State University, 1960.

32. McRuer, D.T., and Krendel, E.S., "Dynamic Responses of Human Operators" USAF, WADC TR56-524, October, 1957.

33. Birmingham, H.P. and Taylor, F.V., "A Human Engineering Approach to The Design of Man-Operated Continuous Control Systems", Naval Research Laboratory, Washington, D.C., NRL Report 4333, April, 1954.

34. Garvey, W.D., and Mitnick, L.L., "An Analysis of Tracking Behaviour in Terms of Lead-lag Errors", Naval Research Laboratory, Washington, D.C., NRL Report 4707, February, 1956.

35. Rund, P.A., Birmingham, H.P., Tipton, C.L., and Garvey, W.D., "The Utility of Quickening Techniques in Improving Tracking Performance with a Binary Display", Naval Research Laboratory, Washington, D.C., NRL Report 5013 September, 1957.

36. Chernikoff, R., Duey, J.W., and Taylor, F.V., "Two-Dimensional Tracking with Identical and Different Control Dynamics in Each Coordinate", Naval Research Laboratory, Washington, D.C., NRL Report 5424, November, 1959.

37. Clymer, A.B., and Ax, A.F., "Possibilities for Human Simulation", ASME Paper No. 60-AV-37, presented at the Aviation Conference, Dallas, June 1960.

38. Bekey, G.A., "Adaptive Control System Models of the Human Operator", Symposium on Adaptive Control Systems, October 1960, Garden City, N.Y.

39. Cosgriff, R.L., and Briggs, G.E., "Accomplishments in Human Operator Simulation", ASME Paper No. 60-AV-40, presented at the Aviation Conference, Dallas, June 1960.

40. Cacioppo, A.J., "Possibilities for Simulation of Human Dynamics and Perception", ASME Paper No. 60-AV-39, presented at the Aviation Conference, Dallas, June 1960.

41. Gold, A.J., "Possibilities for Simulation of Dynamic Physiology", ASME Paper No. 60-AV-35, presented at the Aviation Conference, Dallas, June 1960.

42. Stacy, R.W., and Coulter, N.A., Jr., "Simulation of Human Physiological Systems", ASME Paper No. 60-AV-38, presented at the Aviation Conference, Dallas, June 1960.

43. Clymer, A.B., "Accomplishments in Human Simulation", ASME Paper No. 60-AV-26, presented at the Aviation Conference, Dallas, June 1960.

44. Clymer, A.B., "A Broad Look - Computers and Their Applications, Present and Future", a lecture presented at the Charleston Section AICLE study course on Computers and Their Application to Chemical Engineering, Charleston, W. Va., 15 Dec. 1960.

45. Proceedings of Bionics Symposium", Dayton, Ohio, September 1960 (in press).

TABLE I

VALUES OF COEFFICIENTS DETERMINED
IN MANALOG RUNS HAVING NO ACTUAL
TRIAL-TO-TRIAL VARIABILITY

Trial	Coefficient			
	a	b	c	d
1	+.0473	+.0436	+.2787	-.0075
2	+.0615	+.0547	+.2801	+.0059
3	+.0572	+.0380	+.2781	-.0102
4	+.0575	+.0475	+.2715	-.0062
5	+.0544	+.0337	+.2762	-.0133
6	+.0593	+.0465	+.2813	-.0030
7	+.0557	+.0539	+.2819	-.0002
8	+.0500	+.0420	+.2740	-.0032
9	+.0432	+.0529	+.2757	-.0044
10	+.0614	+.0465	+.2829	-.0109
Average	+.0550	+.0459	+.2770	-.0053
True Value	+.0600	+.0500	+.2800	+.0010

TABLE II

COEFFICIENTS DETERMINED USING
ITERATIVE INITIAL CONDITIONS PROCEDURE

Trial	Coefficient			
	a	b	c	d
1	+.1040	+.0402	+.2077	-.0157
2	+.1815	+.0471	+.2405	-.0033
3	+.1982	+.0499	+.2474	+.0012
4	+.2023	+.0498	+.2486	+.0014
5	+.2011	+.0492	+.2480	+.0012
True Value	+.2000	+.0500	+.2500	+.0010

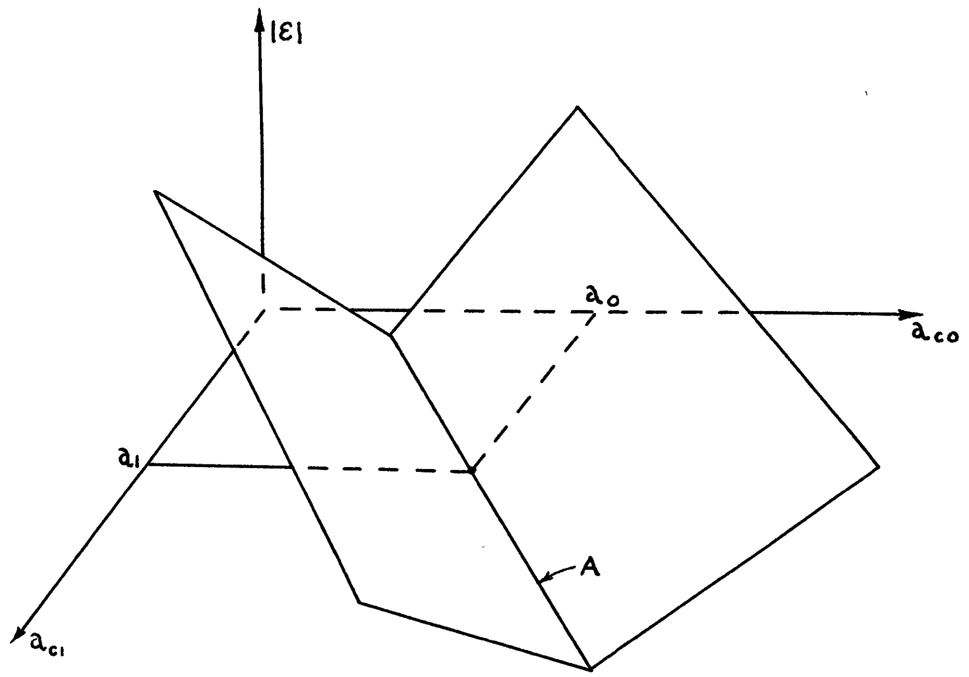


Figure 1 Geometry of Error-Parameter Space

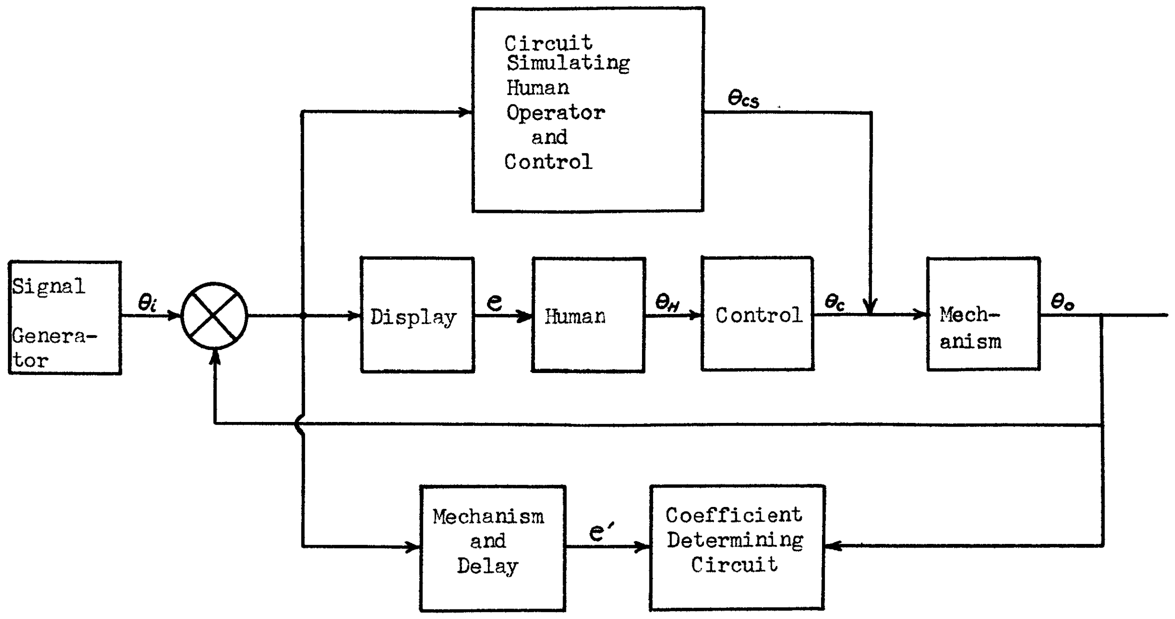


Figure 2 Overall Block Diagram

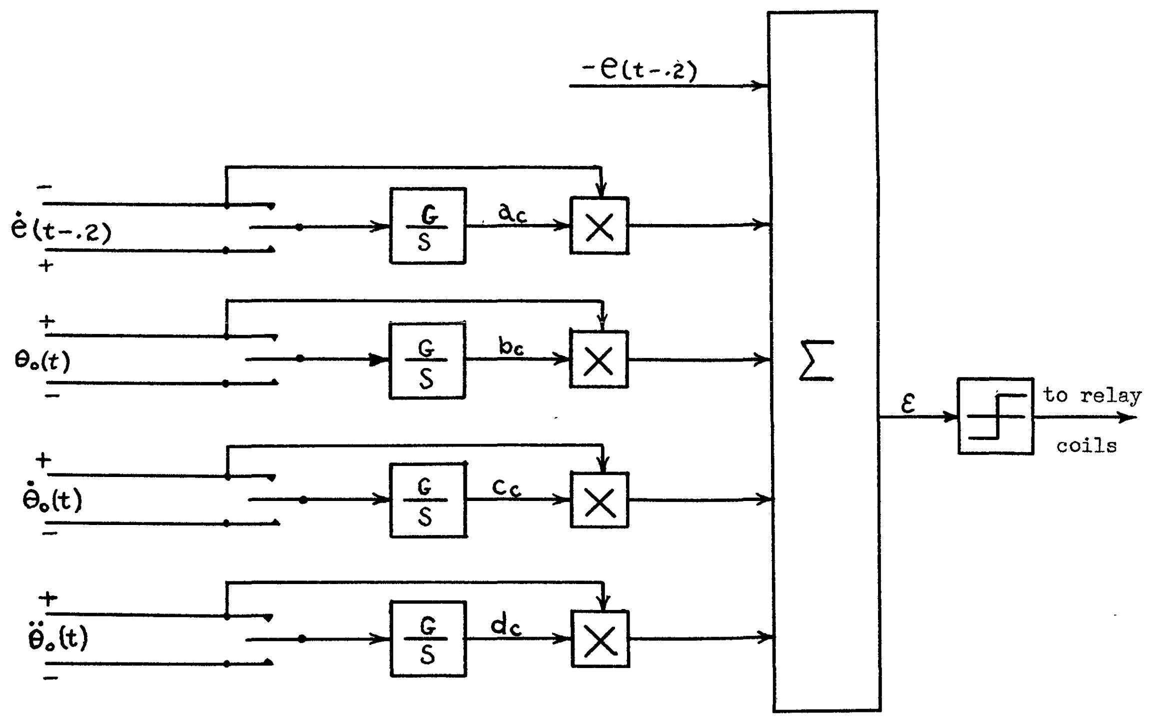
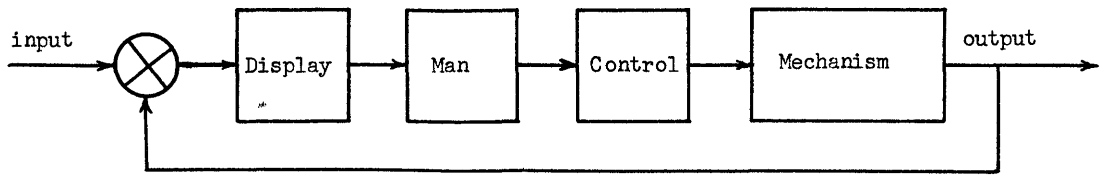
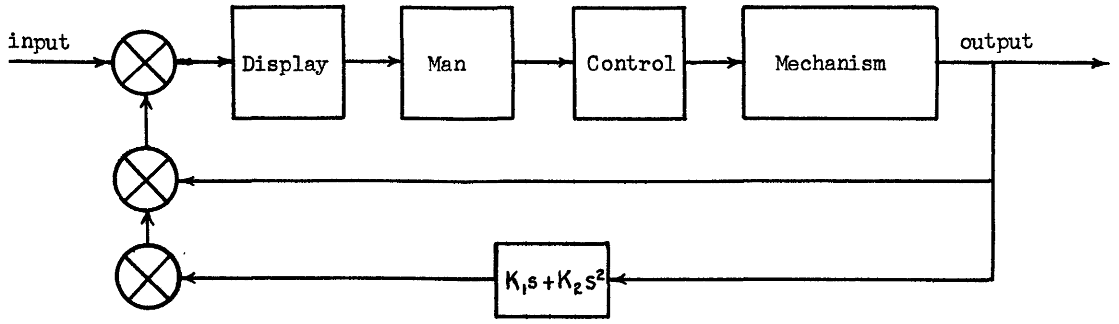


Figure 3 Block Diagram of Coefficient Determining Circuit



A. Unquickened System



B. Quickened System

Figure 4 Example of a Quickened and of an Unquickened System

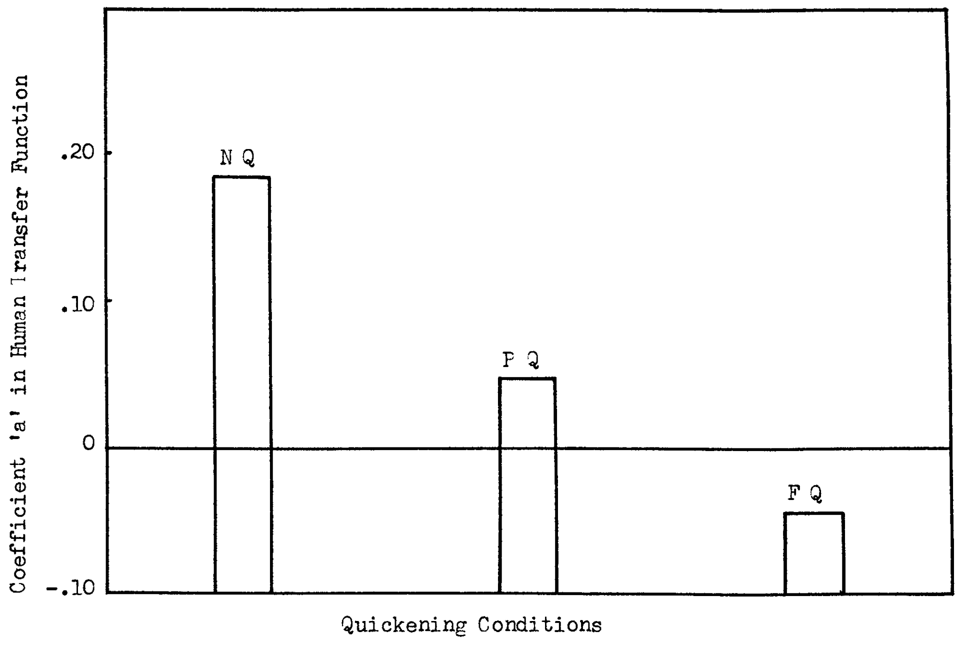


Figure 5 Weights assigned to $\dot{e}$ as a Function of Quickening Conditions

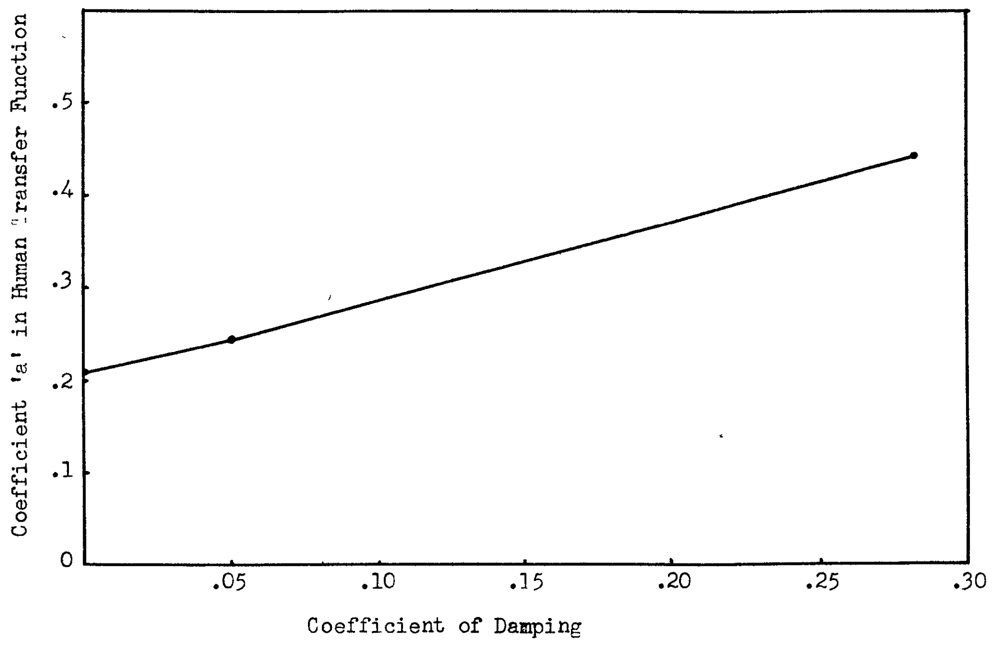


Figure 6 Weights Assigned to $\dot{e}$ as a Function of Damping Coefficient

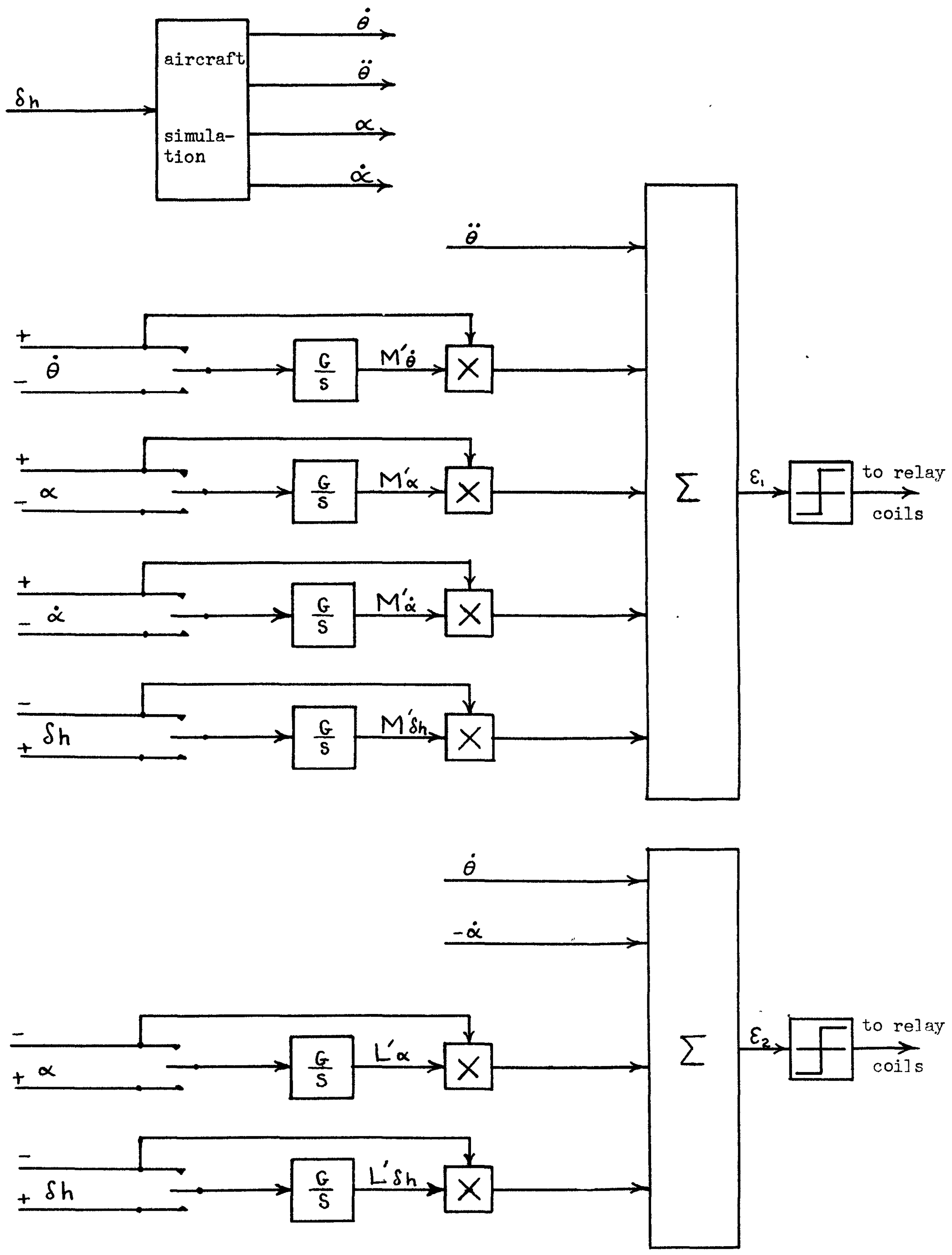


Figure 7 Block Diagram of Apparatus for Determination of Aerodynamic Coefficients

LIST OF EXHIBITORS

Aeronutronic Div., Ford Motor Co.	Newport Beach, California
Ampex Computer Products Co.	Culver City, California
Ampex Magnetic Tape Products	Opelika, Alabama
ANelex Corporation	Boston, Massachusetts
Applied Development Corp.	Hawthorne, California
Automatic Electric Sales Corp.	Northlake, Illinois
Autonetics, a Div. of No. American Aviation, Inc.	Downey, California
The Bendix Corp., Computer Div.	Los Angeles, California
Berkeley Div. of Beckman Instrum., Inc.	Richmond, California
Bryant Computer Products	Walled Lake, Michigan
Burroughs Corp.	Detroit, Michigan
California Computer Products Co.	Downey, California
C. P. Clare and Co.	Los Angeles, California
Clary Corp.	San Gabriel, California
Computer Control Co., Inc.	Los Angeles, California
Computronics, Inc.	Denver, Colorado
Consolidated Electrodynamics Corp.	Pasadena, California
Control Data Corp.	Minneapolis, Minnesota
DATAMATION, F. D. Thompson Public., Inc.	Los Angeles, California
Digital Equipment Corp.	Maynard, Massachusetts
Digitronics Corp.	Albertson, L.I., New York
Dynacor, Inc.	Rockville, Maryland
Electro-Logic Corp.	Venice, California
Electronic Engineering Co. of Calif.	Santa Ana, California
Engineered Electronics Co.	Santa Ana, California
Fairchild Semiconductor Corp.	Mountain View, California
Ferranti Electric, Inc.	Hempstead, New York
Friden, Inc.	San Leandro, California
General Electric Co. Information Systems Dept.	Bethesda, Maryland
General Electric Co. Light Military Electronics	Johnson City, New York
Genesys	Los Angeles, California
International Business Machines Corp.	New York City, New York
Laboratory for Electronics, Inc.	Boston, Massachusetts
Librascope Div., General Precision Inc.	Glendale, California
Monroe Calculating Machine Co., Inc.	Orange, New Jersey
Moxon Electronics	Beverly Hills, California
The National Cash Register Co.	Dayton, Ohio
Pacific Telephone and Telegraph Co.	Los Angeles, California
Packard Bell Computer Corp.	Los Angeles, California
Philco Corp. G & I Group, Computer Div.	Los Angeles, California
Photocircuits Corp.	Glen Cove, New York
Potter Instrument Co., Inc.	Plainview, L.I., New York
Ramo-Wooldridge, a Div. of Thompson Ramo	Canoga Park, California
Recordak Corp.	New York City, New York
Reeves Soundcraft Corp.	Danbury, Connecticut
Remington Rand UNIVAC Div. of Sperry	New York City, New York
Rese Engineering, Inc.	Philadelphia, Pennsylvania
Rheem Semiconductor Corp.	Mountain View, California
Royal McBee Corp.	Port Chester, New York
Soroban Engrg. Inc.	Melbourne, Fla.
Sprague Electric Co.	North Adams, Massachusetts
Stromberg-Carlson San Diego	San Diego, California
Tally Register Corp.	Sèattle, Washington
Telex Corp.	Telex Park, Minneapolis, Minn.
Texas Instruments, Inc.	Dallas, Texas
Uptime Corp.	Denver, Colo.
Walkirt Co.	Inglewood, California
John Wiley & Son, Inc.	New York City, New York