REPORT NUMBER
ORO-2572-27

RICE UNIVERSITY COMPUTER PROJECT

FINAL TECHNICAL REPORT

AEC Contract AT-(40-1)-2572

June 1, 1959 to March 31, 1970

Walter Orvedahl

Walter Orvedahl, Lecturer
Department of Electrical Engineering
Principal Investigator

Submitted June 19, 1970

Table of Contents

Section I

INTRODUCTION

Section II

SUMMARY OF ACTIVITIES

Section III

EVALUATION OF PROGRESS IN THE STATE OF THE COMPUTER ART

RICE UNIVERSITY COMPUTER PROJECT

FINAL TECHNICAL REPORT

AEC CONTRACT AT-(40-1)-2572

June 1, 1959 to March 31, 1970

I. INTRODUCTION

The research proposal on which this contract was initiated was based on work that was supported under an earlier contract. This was AEC Contract AT-(40-1)-1825, which began on April 1, 1957. It was drawn on a proposal submitted by Professor John Kilpatrick. The work involved the production of a large scale high speed digital computer. The primary purpose of the program, as stated in Article 1 of the Contract was "to promote computer research designed to advance the state of the computer art by studies of computer design and components. The role of the computer, when completed, will be that of a regional machine for use by the Contractor to (1) perform research with and on the computer and its components in order to determine new and/or more effective computers or methods, (2) train faculty members, students and others in the operation of computers, and (3) perform miscellaneous computations for research projects of universities and industry."

That Contract provided funds through the period of construction of the computer, which was finished to the point of being usable in 1960. It became known as The Rice Computer. It did include many innovations which had a subsequent impact on the state of the computer art, as will be shown later in this report.

Contract AT-(40-1)-2572, which is the subject of this report, was drawn on a proposal submitted by Professor Martin Graham in April 1959. The initial period of the contract was from June 1, 1959 to September 30, 1960. The primary purpose of this contract was to support a continuation of the research program which was defined in the earlier contract. Professor Graham was senior investigator for most of the period of the earlier contract. He was principal architect of the Rice Computer, and directed every aspect of its production. He has especially high capabilities in the field of electronic design, and he brought a large amount of novelty to the electrical structure of the Rice Computer.

RICE UNIVERSITY COMPUTER PROJECT

FINAL TECHNICAL REPORT

AEC CONTRACT AT-(40-1)-2572

June 1, 1959 to March 31, 1970

II. SUMMARY OF ACTIVITIES

The activities undertaken during the term of the contract are best considered in three periods. The first years of the contract were concurrent with the closing years of the earlier contract, and their objectives were similar. Following that came a period in which the computer and other facilities of the lab were applied to a wide variety of problems. The third period covers the closing years of the contract. These years were devoted to the development of a new arithmetic and logic unit of advanced design.

The first period of activity occupied the time between the beginning of the contract, June 1959, to about October 1962. The activities of this period were directed primarily to getting the Rice Computer system established with all its capabilities, including an initial complement of software. It was during this period that many excellent ideas in the design of both hardware and software were brought forth. The importance of some of them was not fully recognized until much later.

The major components of the computer, such as power supplies and peripheral equipment, were acquired early in the construction program. By 1959 only the control remained to be developed. This was done with the participation of students, who in later years prepared theses related to the work. The installation of the various computational functions was made step by step. The logic functions were completed first, and this enabled Professor Kilpatrick to demonstrate the running computer late in 1959, with a program which solved pentomino puzzles, and represented the solutions diagramatically with the aid of the high speed printer. This was approximately two years after the very beginning of the project, and was probably, at that time, a near record for early success.

Circuit problems related to the computer tape system were examined during the 1959-60 period. The behaviour of parametric amplifiers was studied as a student project independent of the computer construction.

By June of 1961, the computer was committed to scheduled operation, with 8000 words of Central Store, but no magnetic tapes or other back up storage. A software system was completed and put in service. It consisted of an assembler and a set of utility routines which make up an operating system. The concepts involved in this software were influenced in a useful way both by the logic capabilities of the processor and by the tight limitation on available storage. The consequences of this will be described in a later section of this report. An account of the experience gained with this system was published in 1968 [1].

Members of the Project undertook assistance to other laboratories on campus with instrumentation problems. The objective of this was to prepare for the use of the computer for processing data from scientific experiments.

In the following year, the Department of Geology, with assistance from Project personnel, was equipped with a direct digitizing seismograph which recorded information on magnetic tapes readable by the Rice Computer tape system, which was still under development. Early proposals for the use of linear filtering to improve packing density for magnetic tapes proved to be without merit, so conventional amplifiers were constructed.

In mid-1963, the computer tape system was committed to regular operation, for storage of computer data, and for processing seismic tapes. Several other facilities were put into service in that year. A digital to analog converter was built into the computer. Software for generating displays using a 512 x 512 dot raster were written. An ordinary laboratory oscilloscope was connected as a display, for both plotting and alphanumeric displays. The small screen was a handicap for direct viewing. However, a 16 mm motion picture camera was mounted on the oscilloscope, with film advance controlled by machine command. Graphic displays including unusual dynamic effects were demonstrated using this simple equipment.

Laboratory experiments were conducted on the use of a dark trace oscillographic storage tube as a data accumulator for multi-parameter nuclear experiments. The conclusion was that this is not a useful technique.

An algorithmic language called Genie was specified by J. K. Iliffe in 1960, as a part of the Rice software system. A compiler for the implementation of this language on the Rice Computer was developed by Jane Jodeit. This was placed in service during 1963. The compiler was designed to be consistent with earlier parts of the operating system. Dynamic Storage Allocation was included as a normal feature.

Professor Graham introduced a course in Digital Computers. He also served as thesis advisor to Joel Cyprus for his Doctoral dissertation on Optimal Synthesis of Boolean Functions [2].

These activities marked the full maturity of the Rice Computer, and ended the phase of activities which were centered around its development. The work proposed for the next contract year, beginning October 1, 1963 was directed toward a study of the influence of the microelectronics technology on an optimal organization of computer logic. Microelectronic components were then beginning to be introduced commercially.

By the following Summer, samples of digital integrated circuits which were an early form of microelectronic components, were procured and tested. A magnetic core memory was ordered for the computer. The required interface circuits for the memory were under design, using integrated circuits wherever appropriate.

The Project maintained a continuing interest in instrumentation problems which could be aided by digital techniques. An analog to digital converter was constructed and incorporated into the computer. This made it possible to digitize real time waveform inputs or records from instrumentation tapes. In later years extensive work was done using this facility.

An instrument was constructed which could monitor internal processing intervals in the computer, and could display both the number of incidences of a process in unit time, and the percentage of total time attributable to the process in question. Measurements were made on programs running on the Rice Computer, and valuable insights were gained into the utility of fast registers, indirect addressing, etc. [3].

A list processing system was incorporated in the software for the computer, as a by-product of J. A. Robinson's studies in theorem proving. Manuals for the software system were compiled and published as reports.

The work proposed for the year beginning October 1, 1964, contemplated profound changes in the logical structure of the Rice Computer, so that short addresses used for addressing the computer's four word scratch pad memory could initiate an indirect addressing sequence and thus access locations anywhere in the main memory. In addition, it was proposed that an independent address calculator be designed and constructed, which could do the work of the addressing system of. the Rice Computer, but would operate independently with its own control, upon command from the main computer control. This was proposed in part to explore the use of integrated circuits, and in part for the purpose of replacing a significant portion of the Rice Computer with a replacement done in modern circuit technology.

The memories which were ordered the previous year were delivered during the 1964-65 year. Most of the operations of the laboratory were used in preparing the interface circuits for these memories, and getting them into service when they arrived. The logic and circuit design for the address calculator was completed, and a two bit operating model was constructed and tested in the Summer of 1965. Activity in the development of software during that period consisted mostly of the assembler, and the use of relations such as >, <, as binary operators in the Genie language.

For the following year additional studies of changes in the logic organization of the computer were proposed. The use of data tagging on a much larger scale than had been tried in any other computer was contemplated. This included hardware interpretation of the tags, and tag control of the microprogram. Such a logic organization was new, and widely different from that which characterized the old Rice Computer. This study led to the beginning of the third period of activity, in which an entire new processor was.designed and constructed.

The laboratory was occupied, up until September of 1966, with the completion of the address calculator and with the extension of the range of short addresses by additions to the logic of the Rice Computer [4].

A Doctoral thesis entitled "Fabrication and Switching Characteristics of Permalloy Films" was prepared by J. K. Watson who was a staff member on the Project.

In the software area, the most significant development was the extension of the Genie Compiler with the inclusion of a generalized scheme for addressing double word operands. To assist this process, computer modifications were made which would generate the address for a double word operand from a single given address.

The computer was applied to some calculations in statistical mechanics involving multiple integrals. These problems were demonstrated to be completely intractable for systems which allocate storage at compilation time. The Rice storage allocation system was able to handle all examples successfully, without resort to backup storage of any kind [4].

Professor Graham, who had been Principal Investigator and Project Director since the beginning of the contract in 1959, left Rice University in July of 1966. He was succeeded by Walter Orvedahl as Project Director and as co-principal investigator along with Professor J. Alan Robinson.

The Project unexpectedly received the offer of a 750 million bit disc file from UNIVAC in the Spring of 1966, at no cost. This was badly needed as a backup storage, since tapes are unsuitable in scientific computation due to their long seek time. The file was accepted.

The work of the year beginning October 1966 was planned around the task of designing and constructing a controller and data channel as an interface between the disc file and the memory. This was a major construction task. With consideration of future activities of the Project, three boundary conditions were established. They were,

a. The device should serve many peripherals, and not the disc alone.

b. It should be made of the best contemporary components, to stave off obsolescence.

c. It should be electrically compatible with future structures, and should be matched to older structures by special interfacing.

Integrated circuits had been in use and under study in the Project since 1963. By the end of 1966 the laboratory recognized that current switching logic, of a type represented by the RCA ECCSL line, was superior in switching speed, in freedom from self induced noise, and in the versatility and moderate cost of the modules then available. These circuits were adopted as standard for all future work of the Project.

The circuits chosen were available only in the physical form of the flat-pack. This form is not well suited to conventional assembly methods. The Elco Corporation offered a line of materials called the omnicomb system which could be used to assemble flat-packs in closely packed structures with little difficulty, except for the tedium of hand-processing the materials. This proved to be too difficult for most laboratories and the Elco Corporation ultimately discontinued the line.

This laboratory developed a complete set of computer aided design programs which would optimize the placement of parts for the omnicomb process. A special tape controlled punch was made for processing the material. The computer programs produced tapes for controlling the punch, and computer output provided complete documentation for all assemblies. Thus the most time consuming operations were eliminated, and the task of producing a large digital system became tractable for the small force available to the laboratory. Enough material was purchased from Elco to finish all work then projected.

These preparatory operations, and an extensive rehabilitation of the disk drive occupied the laboratory until September of 1967.

The computer was used for some studies in speech recognition by Professor Howard Resnikoff and Mr. G. Sitton [5][6]. The analog to digital converter, installed in 1963, was used for direct digitization of utterances into a microphone. A report entitled "An English Spoken Digit Data Sampler" was prepared and circulated to other investigators in the field [7].

A system of logical memory protection was incorporated into the Rice Computer software. This system checks each reference to a stored data array to assure that it is within the defined limits of the array. The facility can be called upon as needed, as in debugging programs. Since it is slow, it is usually omitted when the program is ready to run. This system was an important antecedent to the future work of designing a new processor.

Mr. John Iliffe, of International Computers, Ltd, London, visited the Project for a period of three weeks in January of 1967. The time was occupied with a thorough discussion of the Basic Language Computer, which was then under construction at ICL [8]. He left a set of notes from which plans were derived for the specification of a new arithmetic processor. Arrangements were made for him to spend the academic year 1968-69 with the Project.

The study of data tags that was initiated in 1965-66 was interrupted to meet the requirement of the disk file controller. It was learned in the discussions of the Basic Language Machine that the concept of Data Tags interpreted by hardware had been adopted for that design. Interest in this concept was thereby reinforced. Accordingly, the work proposed for the period October 1, 1967 to September 30, 1968 included an implementation of a system of data tags on the Rice Computer as part of the work of the laboratory. It also contemplated the preparation of specifications for an entirely new computer, or at least a new logic organization, based on concepts worked out together with John Iliffe. It was recognized that with the new integrated circuit technology, it would usually be more practical to construct large portions of the Rice Computer anew, than to interface new functional parts to the old. Accordingly the option to replace the old computer in its entirety was held open.

Before the above work could begin, construction of the Disc File controller and data channel had to be completed, using the techniques developed in the previous year. This took until June of 1968.

Production facilities were then committed to construction of of a modern counterpart to the 54 bit arithmetic registers and adder which comprise the arithmetic and logical unit of the Rice Computer.

Upon completion of the controller, the disk file was placed in operation. To do this, a long period of fault reduction was required over the integrated circuit structures that had just been fabricated. This was a lengthy process, because there were many subtle hazards about the assembly techniques that were not appreciated at the outset. For example, circuit pins were hand soldered one at a time, and then trimmed to length. In some cases soldered connection was confined to the tip region only, and trimming removed it, leaving an open circuit. Dip soldering was adopted for all future assemblies. Because of these delays, successful operation of the disk was not acheived until November of 1968.

Other work was accomplished during this period. Much of it was directed to preparation of the system for time sharing operation. An IBM Selectric input writer and Tektronix 611 display storage scope were added to the system. The display utilizes the computer memory and digital to analog converter for writing the 1024 x 1024 dot raster.

In software development, an interpretive Formula Evaluator was written. This is suitable for conversational use of the computer in the manner of a desk calculator, but it also includes more powerful capabilities such as program looping, and calls to library routines. This work was performed by Mr. Grant Youngman, who was awarded a professional Master of Electrical Engineering degree in the Spring of 1968. He received his commission in Naval ROTC and entered a four year tour of active duty.

An addition was made to the Genie Compiler which generalized implicit operations by applying them to subscripted matrices. This greatly simplifies program loops which involve operations on matrices within the loop [9].

The research of Professor Resnikoff and Mr. Sitton in the area of speech recognition was continued. The work led to a suggested prosthetic device for speech recognition in cases of total loss of hearing in the high frequency range [10].

The work proposed for the period October 1, 1968 to September 30, 1969 again called for completion of a new processor oriented toward the use of data tags. By this time the processor had been partly specified, but production work had begun only on portions that involved no new architectural concepts. The electrical performance, logic design, and physical structure was, of course, consistent with the most advanced contemporary practice.

The designers had been urged to preserve the logical form of the old Rice Computer, in order to avoid the nullification of existing programs. As a concession to this important consideration, it was agreed that the processor would have two control systems, and two vocabularies, one being a reproduction of the old Rice Computer. This plan was abandoned when it was recognized that performance of the new system could be so high that interpretive procedures could duplicate the logic of the old Rice Computer, and match it in computing speed.

Mr. John Iliffe joined the Project in September 1968, as a visiting professor of Computer Science. Other members of the Project staff joined him in development of a complete specification of the architecture of the new processor. The Data Tagging Concept permits a rather simple vocabulary set. However, the internal logic of the computer is unusually complicated, since a great many things in this computer are done by hardware which would normally be done by software. The detailed specifications of the internal logic has been done by Dr. Sigsby Rusk, Chief Engineer and logic designer. This work was undertaken after the architecture was specified.

The circuit assembly process proved to be efficient. All attempts to fit portions of the old computer into the new architecture were extremely wasteful of time, since they involved working out special procedures instead of using the well automated and computer optimized assembly processes used with new construction.

Accordingly, the decision was to make the new processor in its entirety from new components, and leave the old processor intact, to play its role in the production process until no longer needed.

Assembly of the circuits was about 90% completed by September 1969. There remained a large task of backplane wiring, for which the details remained to be specified. After that, there remained the fault reduction, or debugging process over the entire structure. A final renewal of the Contract was granted for the period October 1, 1969 to March 31, 1970, for the specific purpose of completing this work. It was provided in the Renewal Proposal that if any work remained to be finished at the expiration of this period, Rice University would undertake to see it through to completion.

That arrangement satisfies the aims of the research program fully, but perhaps a comment is in order about the apparent lack of rigor in estimating completion dates. The reason for this is the fact that this undertaking, like others that have been done before, at Rice University and elsewhere, are exploratory in nature, and the product is really a prototype. The major part of the work lies in the specification of minute details, such as choices of components and preparation of wire connection tables. The magnitude of this task cannot be known precisely until it has been completed. In order to estimate construction time accurately, the beginning of the construction process would have to be delayed until all specifications were completed. This would greatly extend the total time required for the task, and it would be an unnecessary delay, as it is entirely natural and feasible in this type of work to begin construction when only a part of the device has been completely specified. That policy was followed in this case. The assembly process has always been able to keep pace with the preparation of the specifications, so the total time is essentially equal to just the time required to generate the specifications.

As of the time of writing this report, the last of the backplane wiring is being installed. The debugging process has been partly accomplished over sub-assemblies during the last few months. It is scheduled to begin over the entire system beginning July 6. The University is committed to placing the computer in service, and the computer is to play a central role in the academic program in Computer Science.

RICE UNIVERSITY COMPUTER PROJECT

FINAL TECHNICAL REPORT

AEC CONTRACT AT-(40-1)-2572

June 1, 1959 to March 31, 1970

III. EVALUATION OF PROGRESS IN THE STATE OF THE COMPUTER ART

During the latter half of the 1950 decade, new generations of computers were under development in many industrial laboratories and in a few universities and a few national laboratories. Many of the highlights of the developments of this period are summarized in reference #11 which may serve as a bench mark for the state of the computer art near the beginning of the term of this contract.

The most important advances in the area of computer components during this period were the universal adoption of transistors and crystal diodes as logic elements, and of ferrite cores as memory elements. In early applications, these elements were sold with the burden of development cost included in the price. They were, however, inherently cheaper than their earlier counterparts, and were much more reliable. These two factors were fundamental to the success of the more elaborate logic structures which the new computers introduced.

The Rice Computer was a product of that period. However, most of its planning was done prior to 1957, so it did not gain the benefit of the revolution in components, and there was little in its physical structure which had a direct impact on the state of the art, although some concepts are indirectly relevant.

Error Correction System

The computer was equipped with an electrostatic storage tube memory. This has an important, though indirect, relevance to contemporary computer structure, in the sense that the storage tube represents an early example of large scale integration, with its attendant problems.

The storage tube is equivalent to an array of as many as 16,000 binary storage elements, structurally integrated. If a single one of the elements in defective, either the entire structure has to be thrown away, or means must be found to live comfortably with no help from the defective element. A single storage tube is worth about $400.00. The probability that it will have at least one bad element is a virtual certainty. Thus throwing it away is not an acceptable solution. This simple statement of the problem would be familiar to people who are working to develop large scale circuit integration today.

The solution, in the case of the electrostatic memory, was to include additional storage for an error correcting code, and to provide error correction in the system. Since the memory, including error code storage occupied 63 parallel storage tubes, the error code system added a factor of one-eighth to the storage tube requirement. It achieved an essentially perfect solution to the problem of defective storage elements, since the probability that defects in more than one tube at corresponding locations in a system of this size is 1/256.

Professor Martin Graham, at the University of California, Berkeley, who was the designer of this system, has examined the relevance of this approach to modern large scale integration techniques. His findings were presented in a seminar at Rice University in 1968. We do not have a published record of them. Qualitatively, he was able to show that, for arrays of a suitbale configuration, the inclusion of extra elements for the error correction function could dramatically reduce the rejection rate, or alternatively, could greatly augument the permissible number of elements in a single array, while holding to an acceptable rejection rate.

Architectural Features of the Rice Computer

The instruction format of most computers is arranged so that each instruction includes a single elementary operation to be applied to one or two operands. This is the arrangement that provides the greatest generality. If the computer word length is short, instructions are often packed one per word. If it is longer, they may be packed two per word.

The Rice Computer has a 54 bit word, which is unusually long. Nevertheless, instructions are packed one per word. This is because instructions in the Rice Computer are compound expressions, with as many as four operations included in each. Only one of these operations may be chosen from the full vocabulary of the computer. The others are chosen from an extremely limited set. If the available associated operations do not fit the needs of the program at a particular point, they are coded as null. In such a case the instruction accomplishes only one elementary operation, and the storage used for the null portion of the instruction is in some sense wasted. Clearly, such an instruction format has unfavorable properties. It is acceptable only if there is a high probability that several of the operations in each instruction will be useful most of the time.

This turns out to be the case for all programs which have been optimized either by the compiler or by the efforts of a programmer working with an assembler.

The operations available are:

1. Formation of the magnitude, or the negative of an operand held in an arithmetic register, an index register, or any of four scratchpad registers, and loading the

result into the accumulator register.

2. Formation of the above, applied to any operand anywhere in store, including any of the above registers, and loading the result into the second operand register.
3. Perform any operation in the vocabulary set, on the two operands addressed in 1 and 2.
4. Store the result in any of the above registers, or alternatively, increment an index register.

In practice, operation 4 is useful in a majority of cases. Operation 1 and 2 are useful in special cases. The greatest usefulness of the multiplicity of operations comes when the computer operates in repeat mode. The index register in 4 above is automatically tested for zero in every cycle of execution. Data tags are also tested. The instruction being repeated while in the repeat mode may be a test instruction. The repeat sequence terminates when any of these tests are satisfied. With these features any one of several operations can be applied over an array of any length using a single instruction, and with no instruction fetching. Such a procedure would require a program loop of several instructions of ordinary form. Usage has shown that the Rice system is economical in its instruction storage requirements, and its speed of computation is higher than is expected on the basis of its electrical response times.

The data tags mentioned above have been used in only a few other computers contemporary with the Rice Computer [12][13]. In the Rice Computer arithmetic and logical entities are 56 bits long. The two extra are interpreted as a set of three markers or tags. The 00 configuration is interpreted as an untagged word. By means of these tags, selected locations in memory may be marked. The tags must be written or erased by explicit instructions. When a word occupying a tagged location is moved to a new location, the tag is not moved.

When operands or instructions are brought into the processor, the tag bits of value 1 are entered into a tag register. Tag bits of value zero are not entered. Thus the tag register accumulates the "or" of all tags encountered. The tag bits may be individually tested for either value 1 or value 0, and a control jump can be taken on the basis of the test.

Alternatively, the computer may be operated in trapping mode. The state of the tag register, or of any of several other indicators in the computer, such as overflow, can cause a trapping sequence to be initiated. In this computer the sequence begins with an unconditional jump to a fixed memory location assigned to serve the particular trap condition encountered. This location will be preloaded with a jump to a program designed to respond to the trap condition. Each of the conditions which can cause a trap can be suppressed or enabled by a mask bit held in a special register.

Control of the computer by means of the data tags is an elegant and efficient system. When operated in trapping mode, it permits the testing procedures required for control to be accomplished in parallel with other processes in the computer, so they involve no loss of time. Moreover, since the tags are independent of any other portion of the data structure in the computer, a control regimen may be superimposed on any program, at will, without altering the logic of the program. This is especially useful for putting a program through the several phases of debugging that are often required.

The utility of data tags has been well enough proven so that the exploration of a more elaborate scheme of data tagging became a major research effort in later years.

A complete expression for a memory address in the Rice Computer Instruction Format occupies 24 bits in the instruction word. Fifteen bits are used to designate the basic memory location number. Eight bits are used as selectors for each of eight index registers. The effective memory location number is the sum of the basic location number and the content of all index registers selected.

The one remaining bit is used to designate the indirect addressing option. When it has the value 1, the 24 bits in the address portion of the instruction register will be replaced by the corresponding 24 bits in memory at the location expressed by the present effective location number. Since the entire address expression is replaced in the process, indirect addresses can be chained, with indexing at every level.

Operation codes are provided in the vocabulary for loading, incrementing, and testing individual index registers. The registers are addressable as operand locations, so the content of any of them can be brought into the arithmetic processor and used as an operand, or it can be stored in memory.

Two of the registers are dedicated to important hardware control functions. One of these serves as the instruction sequence counter. This makes it possible to write programs that are location independent, by selecting this counter as a component of every operand address. The other dedicated register is called the Pathfinder. Whenever a control jump occurs, the Pathfinder receives the present content of the counter. It thus maintains the essential information for a return to the program exit point. This may be stored in memory, or it may be used directly by selecting the Pathfinder as a component of the address in the return jump. The concept of making the program counter and pathfinder both members of the index register set was originated in the Rice Computer design. It was one of the earliest efforts to relieve the software system from the overhead burdens in program control.

Operating System Software

When a computer with a new vocabulary set is placed into operation, the first thing that is required is a set of translators which will establish correspondence between the barely intelligible coding of machine instructions, and a mnemonic symbolism more comprehensible to humans. The most primitive parts of the translators must be done in machine language. The finished program is called an assembler, and the mnemonic format for instructions is called an assembly language. Usually, one line of assembly language code generates one computer instruction, including the assignment of addresses. Along with the assembler, a collection of utility routines must be prepared as soon as possible to handle initial loading of the computer, input and output processes, editing, diagnostics, etc. This collection is called the operating system, since these programs must be resident in memory during much of the time that the computer is running.

The elaborate addressing facility of the Rice Computer made it attractive to incorporate some advanced ideas into the operating system. As suggested above, the assignment of storage space is one of the functions of the operating system, usually handled by a routine called a loader. The designers of the Rice operating system, Mr. John K. Iliffe and Jane G. Jodeit, undertook to write a completely general storage allocation routine. The routine is called STEX, a mnemonic for storage exchange.

The STEX routine operates not only at load time, but can be called upon at any time during the running of a program, and can assign storage to a process as the need arises. Assignment is usually made in blocks of a size specified by the calling program. When space is no longer needed, STEX may be called to return the block to the status of available storage. As might be expected, the latter process can result in a checkerboard pattern of unoccupied blocks. STEX can search this space for a block of a required size. If none is found, it can make a total re-organization of storage assignments, consolidating all unused space into a single block at the high end of memory, from which further allocations are then made as required.

A procedure of this sort is known as dynamic storage allocation. The particulars of this system were published in October 1962 [14]. Dynamic storage allocation routines have probably been applied to special problems many times over the years. The STEX system is one of the earliest to be incorporated into an operating system and to be universally used in regular computation.

The ability to allocate storage as needed, eliminates the waste that results if storage must be reserved on a standby basis at compile time, because it is known to be required at some phase of operation of the problem. This saving has remarkable significance in the economy of the computer. The Rice Computer was able to operate successfully for years with only 8K words of storage, at a period when the normal complement was 32K words. In one instance, a problem in statistical mechanics was undertaken at Bell Telephone Laboratories, which involved operations on a multiplicity of matrices whose dimensions could not be determined in advance. By standard practice, storage was reserved for each of them at the maximum dimension that could possibly occur. Inevitably a case would arise in which all available storage in a 32K system was insufficient to meet the requirement. This problem was moved to the Rice Computer. Due entirely to the dynamic storage allocation system, all cases of the problem were run successfully with only 8K words of storage [17].

The system as published by Iliffe and Jodeit [14] had a wide ranging impact on computing practice in later years. The STRESS system, written by the Department of Civil Engineering, MIT includes a storage allocation routine which is an adaptation of the STEX system. The authors credit the work of Iliffe and Jodeit, but references are not included in the STRESS manual. One of the authors, Professor Robert D. Logcher, included acknowledgments in a subsequent publication [15].

The STEX system influenced the design of software for project MAC, at MIT during the same period, and was acknowledged by the authors [16].

The STEX system as implemented provided a natural way of organizing data in multidimensional arrays, and was appealing as a means of keeping track of where things are stored, quite apart from the fluctuating nature of storage requirements. By means of this system a generalized method of referencing arrays was devised [9]. A storage protection system was incorporated into the Rice operating system which operates on the principle of bounds checking for arrays. Finally, the properties of this scheme of memory organization has become the basis for a new concept in computer organization wherein many of the programmed procedures for implementing STEX are wired into the hardware.

An Algorithmic Language for the Rice Computer

The designers of the Rice operating system software also defined a high level algorithmic language, which was named GENIE [18]. The effort of the designers was not directed to the specifics of the language itself, but to the design of a formal system in which the language could operate. The designers recognized that the language itself should best be specified by the users, to suit their own needs. That is a concept well appreciated today, but at the time GENIE was reported in the literature, efforts were directed toward a "universal language", and any proliferation of languages was regarded as pure mischief.

A compiler for GENIE was implemented on the Rice Computer between 1961 and 1963, and elaborated thereafter. It has been used by all who have prepared problems for the Rice Computer. The feature which seems to be most highly appreciated is the provision for coding operations on arrays. The operations of matrix algebra can be expressed in the same form that is familiar in hand computation [9]. This feature was made possible because the storage control procedures mentioned in the previous section are also provided in the GENIE compiler. Another feature which is much appreciated is the fact that programs in GENIE are compatible with programs written in assembly language.

The GENIE language is closer to traditional mathematical notation than most computer languages are. For example, all input keyboards have a special superscript and subscript mechanism. Codes are associated with these vertical movements of the typescript, and they are interpreted by the compiler as they are in ordinary math. GENIE resembles the MADCAP language in this respect [19].

The stated objectives of the designers of the GENIE system were to generalize a formal system which could be used to describe both numerical and analytical processes in the same system. The concept of evaluation was a dominant one. It was intended that the system should recognize when everything necessary for evaluation of an expression has been specified, and it should then perform the evaluation and replace the expression by its simpler form.

The actual implementation of the compiler was a lengthy task of assembly language programming, and there was some urgency toward placing this high level language into service as soon as possible, on any terms. Accordingly, the translators were devised in the simplest way. The subtleties required to meet the concept of continuous evaluation were not included [20]. The intended generality of the system evolved slowly, with the addition of a few new functions each year.

The system was never implemented on any other computer, and therefore it had no chance of being adopted, like STEX, as a standard software technique. It is probable that some of its basic ideas, such as the specification of context, have been included in the structure of many modern compilers. It is also reasonable to consider that some of the concepts, such as continuous evaluation, are worth some exploration as a matter of research, free from the urgency of the task of preparing needed software.

Investigations in Computer Architecture

Although the Rice operating system established precedent in dynamic storage allocation which were vital to later time-sharing systems, the Rice Computer was completely lacking in all of the control facilities needed to accommodate time sharing. Plans were made to correct this deficiency by additions to the logic structure of the computer. The task of installing the necessary components proved to be uneconomic, because too much hand labor and detailing was involved.

Other aspects of the computer were re-examined, and some extensions were made. In particular, the use of short address fields to select operands from a small set of registers was expanded by adding a special tag bit on each of a set of four of the registers. The tag was interpreted by the control to mean that when set to 1, the register contained, not an operand, but the address of the operand [4]. By this means each operation associated with a short address could be applied to any operand in storage. When this modification was completed, some existing programs were updated to make use of it, and reductions of about 10% in running time were noted in some cases.

With the extended addressing facility, plans were discussed for adding to the vocabulary a larger choice of compound operations, so that such things as the formation of inner products could be done by a single instruction in repeat mode. No additional operations were implemented, because it was recognized that the gains in performance that are possible, although spectacular, can seldom be realized in compiler generated code. The efforts of the Project were therefore diverted to aspects of computer architecture that could be of more general interest.

An instrument for making measurements internal to the computer control was introduced in 1964. It could measure either the percentage of total time spent in some control state, or the total number of instances of a control state during an interval of time. This instrument was used in an experiment conducted by Mr. J.K. Iliffe in the Summer of 1964, to evaluate the usefulness of certain features of the Rice Computer, particularly the four word scratchpad fast memory, the set of index registers, and the use of indirect addressing, which forms the basis of the storage allocation and control scheme used in the operating system.

The important conclusions from the study indicated the following:

1. Indirect addresses account for only a small percentage of memory accesses and are therefore worthwhile, since they are fundamental to the storage control system, and any alternative would probably involve a larger percentage.
2. That fast stores and index registers, whose use has to be pre-planned, impose an overhead burden just in the preloading process. Moreover, in order to gain the benefit from them, the planned usage must be rational. It would be useless to pre-load them just to accommodate one sequence of accesses. It is very difficult to get rational

planning from a compiler, and without it these registers can do more harm than good.

Mr. Iliffe therefore recommended an autonomous loading arrangement, for a fast memory such as the slave store in the Atlas-Titan system, which does not incur any program overhead in its operation. These conclusions and other observations made it evident that many architectural concepts in the Rice Computer should be recast. A replacement of the processor seemed more attractive from a research point of view than any alterations that might be made in the existing structure.

One feature of the Rice Computer that seemed to merit further development was the system of data tagging. This feature requires a memory word that is longer than the arithmetic word. The Rice Computer is equipped with memories of a word length of 64 bits, to accommodate the error correcting codes mentioned in the first part of this report. With modern memories, which were acquired in 1965, the error correction is no longer needed. Thus as many as eight additional memory bit positions became available for experimentation.

Plans for a successor to the processing unit of the Rice Computer were discussed beginning in 1967, with the participation of Mr. Iliffe, who visited the Project for 3 weeks at the beginning of the year. He joined the Project for the academic term

September 1968 to June 1969. During that period, the general specifications of the new Rice processor R-2 were established. This processor will be placed in service in Autumn 1970.

The first consideration in the design of the processor was the recognition that data stored in memory usually occurs in sets which should be structured in a way that is favorable, at least conceptually, to the processes that are to be applied to it. Examples are vectors and matrices, linked lists, tree structures, etc. Programs must include procedures for mapping the desired structure onto the structure of the store, which is usually just an enormous linear sequence of consecutively numbered storage cells.

The architecture of the computer should have in it features which aid this mapping process, allowing it to be done with shorter program sequences. For elaborate structuring, an expression for a memory address must include several components. A separate calculator section for evaluating such expressions to form effective addresses will earn its cost with a high usage factor.

A second consideration, closely related to the first is the fact that all computing involves a multiplicity of data types. Maintaining correspondence between operation types and operand types is not only a burden to the programming process, but it imposes constraints on data structures as well.

To deal with this problem, it was proposed that the extra memory positions, external to the arithemtic word, be committed as tags to identify each word of data as to type. This would, first of all, permit data structures of any form, since the data type could be determined from the associated tag, and not from its location in store. Therefore arrays of mixed type were permissible.

As a corollary benefit, once data can be identified as to type it is no longer necessary to use function codes which are specific to the data type. A single code for an operation, such as +, ×, ÷, etc. has been included in the vocabulary. The data type can be recognized from the tag, and the execution algorithm will branch to paths through the microprogram appropriate to the data type.

A third consideration was the fact that with the more elaborate control procedures required for structuring data, certain forms of stored words would need to be defined for use in control. For example, arrays would need to be associated with words describing the form and size of the array. Moreover, such words would need to be secure from alteration due to misdirected operations arising from program faults. The tagging system, which can guide some operations, as noted above, can prohibit them altogether if they are directed to operands which are really words defined for control.

The specifications as completed define a processor with a shorter and simpler vocabulary than most contemporary computers use. Sixteen different data types may be distinguished by tags that are associated with individual words, and are interpreted by hardware. Two additional tag bits are included for interpretation by software as was done in the first Rice Computer. Of the sixteen data types, eight are reserved for maintaining control in a program, defining arrays, etc.

The matching of operations to operand types is a lesser task than it has been in the past. In any case, an inappropriate match will be detected by hardware as it appears, and corrective action will be taken. Words which are used for control purposes cannot be altered by inadvertently using them as operands. Words which define arrays include information about the array boundaries, and all references to arrays are subject to a hardware bounds check. Any error will initiate corrective action.

The distinctive features of the computer have been detailed in three papers which have been submitted for publication to Communications of the ACM [21][22][23]. One describes the tag system, a second discusses addressing and the structuring of data, and the third describes a new and powerful concept for operating on data by the use of a stack. Preprints are included with this report.

A logical equivalent of the new processor has been simulated on the old computer. System programs have been written, de-bugged, and put through actual use using this simulator. System programmers report that codes are easy to prepare in the mnemonic assembly language of the new processor.

The compiler BCPL is being implemented for the new processor through a bootstrap technique, starting with a version of this compiler written for a GE635 computer. All future system software for the new processor will be written in BCPL. The bootstrapping operation is to be conducted as a part of a general study of methods for the interchange of programs between dissimilar computers. This work is being conducted at Rice under AEC Contract Number AT-(40-1)-4061, with Professor Edward Alvin Feustel as Principal Investigator.

Reference

[1] Jodeit, J.G., "Storage Organization in Programming Systems." Comm. of the ACM, Vol. 11 #11, Nov. 1968.

[2] Cyprus, J., "Optimal Synthesis of the Boolean Functions of Four Variables with Majority Logic" (thesis). Rice University Studies, Vol. 50 No. 2, Spring 1964.

[3] Tung, C., "Operation Time Monitor for a Digital Computer" (Thesis). Rice University 1964.

[4] AEC Report No. ORO-2572-1, (Progress Report 1966).

[5] Resnikoff, H. and Sitton, G., "Linguistic Segmentation of Acoustic Speech Waveforms", Proceedings of the Seventy-Fifth Meeting of the Acoustical Society of America, Ottawa, Canada, 21-24 May 1968.

[6] Sitton, G., "Acoustic Segmentation of Speech", (Thesis) International Journal of Man-Machine Studies, Vol 2 #1, March 1970.

[7] Resnikoff, H. and Sitton, G., "An English Spoken Digit Data Sampler". AEC Report No. ORO-2572-13.

[8] Iliffe, J.K., Basic Machine Principles (book), Elsevier Press 1968.

[9] Sitton, G., "Operations on Generalized Arrays with the Genie Compiler", Comm. of the ACM, Vol 13 No. 5, May 1970.

[10] Resnikoff, H. and Sitton, G., "A New Type of Hearing Aid". Rice University Review, Fall and Winter 1968.

[11] Beckman, F.S., Brooks, F.P., Lawless, W.J., "Developments in the Logical Organization of Computer Arithmetic and Control Units", Proc. IRE Vol. 49 No. 1, Jan. 1961, P. 53.

[12] Gram, C. et al, "Gier, A Danish Computer of Medium Size" IEEE Trans. ED-12, Dec. 1963.

[13] RCA 601 Computer System Manual.

[14] Iliffe, J.K. and Jodeit, J.G., "A Dynamic Storage Allcoation Scheme". The Computer Journal, Vol 5 No. 3, October 1962.

[15] Logcher, Robert D., "Dynamic Memory Allocation for Engineering Data", MIT Report T-66-1, Jan. 1966.

[16] Dennis, Jack B., "Segmentation and the Design of Multiprogrammed Computer Systems", J. of the ACM, Vol. 12 No. 4, p. 589, Oct. 1965.

[17] Rudd, W.G., Salsburg, Z.W. and Masinter, L.M., "Evaluating N-Dimensional Integrals with Polytope Bounds", J. of Comp. Phys., Vol. 5 No.1, p. 125, Feb. 1970.

[18] Iliffe, J.K., "The Use of the GENIE System in Numerical Calculation", Annual Review in Automatic Programming (book), R. Goodwin, Ed., Pergammon Press 1961.

[19] Wells, M.B., "MADCAP, A Programming System for MANIAC II", Annual Review in Automatic Programming (book), R. Goodwin, ED. Pergammon Press 1961.

[20] Iliffe, J.K., "Continuous Evaluation", Introduction to System Programming (book), P. Wegner, Ed., Academic Press 1964.

[21] Orvedahl, W., Iliffe, J.K., Rusk, S.K. and Sibert, E.E., "Hardware Aids to Classification of Data in Storage", AEC Report ORO-2572-24.

[22] Orvedahl, W., Iliffe, J.K., Rusk, S.K. and Sibert, E.E., "Storage Control and Addressing for Complicated Data Structures", AEC Report ORO-2572-25.

[23] Orvedahl, W., Iliffe, J.K., Rusk, S.K. and Sibert, E.E., "Automatic Stacking for A Computer with Explicit Addressing", AEC Report ORO-2572-26.