SEL-76-028

Research in the Digital Systems Laboratory: August 1975 - July 1976

July 1976

Technical Report No 123

Sponsors of research described in this report include the Joint Services Electronics Program, the National Science Foundation, the National Aeronautics and Space Administration, the Air Force Office of Scientific Research, the Energy Research and Development Administration, the Army Research Office-Durham, and the Adva nced Resea rc h Projects Agency. Specific grants and contracts are cited in each research section.

DIGITAL **SYSTEMS LABORATORY**

STANFORD ELECTRONICS LABORATORIES

STANFORD UNIVERSITY . STANFORD, CALIFORNIA

SEL 76-028

RESEARCH IN THE DIGITAL SYSTEMS LABORATORY:

AUGUST 1975 - JULY 1976

Technical Report No. 123

July 1976

Digital Systems Laboratory
Departments of Electrical Engineering and Computer Science
Stanford University
Stanford, California 94305

Sponsors of research described in this report include the Joint Services Electronics Program, the National Science Foundation, the National Aeronautics and Space Administration, the Air Force Office of Scientific Research, the Energy Research and Development Administration, the Army Research Office-Durham, and the Advanced Research Projects Agency. Specific grants and contracts are cited in each research section.

RESEARCH IN THE DIGITAL SYSTEMS LABORATORY:

AUGUST 1975 - JULY 1976

Technical Report No. 123

Digital Systems Laboratory
Stanford University
Stanford, California 94305

ABSTRACT

This report summarizes the research carried out in the Digital Systems Laboratory at Stanford University during the period August 1975 through July 1976.

Research investigations were concentrated into the following major areas: Computer Performance; Computer Reliability Studies, including fault-tolerant computing, evaluation of dual-computer configurations,and implementation of reliable software systems; Computer Architecture, including organization of computer systems, feasibility of real-time emulation, and directly executed languages; Design Automation of Digital Systems; Computer Networks, including network interconnection protocols, the 2000 terminal computing system, and packet-switched network technology/cost studies; LSI Multiprocessors; Compiler Implementation; and Parallel Computer Systems.

TABLE OF CONTENTS

page no.

page no.

DIGITAL SYSTEMS LABORATORY RESEARCH

August 1975 - July 1976

A. COMPUTER PERFORMANCE

E. J. McCluskey

JSEP Contract N00014-75-C-0601

A study has been undertaken to gather and use data on computer system performance to improve the efficiency and reliability of computer architecture and complex computer program systems such as operating systems.

A set of trace programs has been developed to gather data on computer performance. These programs provide data on operating system and user performance for IBM Systems 360 and 370 computers. The trace programs are unique in their ability to gather data on operating system functions such as supervisor calls, and input/output operations, and also in their ability to collect real as well as virtual addresses. The trace data is being used for a number of purposes, including validation of models of computer systems, evaluation of new computer architectures, and studies of the effect of program modularity, interfaces, and primitive operations on system performance.

The same techniques can also be applied to the measurement of small scale processors, specifically microprocessors, to gain a detailed understanding of the architectural issues they give rise to. The basic characteristics of microprocessors can be analyzed with respect to usability, cost, and performance with a study based on the measurement of actual systems. Thus, the empirical results provide the framework in which the fundamentals of microprocessor architectures can be understood.

1. Trace Facility
(D. J. Rossetti)

A general trace facility has been implemented that traces instruction-level operation of the IBM System 370 running with the OS/VS2 operating system. Parts of the operating system are modified dynamically to allow the trace program to gain control at the signal of a hardware attachment, called the Trace Ace. The Trace Ace is a special hardware device developed specifically for this project. Once in control, the trace program records the execution of up to approximately 4000 consecutive instructions beginning with the instruction following the hardware signal. The information gathered is transferred to an external storage medium such as tape or disk storage. After tracing the instructions, the trace facility goes dormant until the next signal from the Trace Ace, The trace sample is limited to about 4000 instructions to keep the main storage buffer space at a reasonable size and to avoid excessive domination of processor time due to tracing overhead.

The Trace Ace is a variable rate periodic timer used to provide the external stimulus required to obtain the trace data, Through a simple hardware modification, the timer signal closes a switch and causes an interrupt. The interrupt is recognized by the trace program, which begins the operation. The Trace Ace then resets for the next time signal or tick. The interval between ticks is determined by a setting on the Trace Ace. The interval can vary between one second and a few minutes.

The tracing system has been demonstrated to be stable, and it has been used to create trace tapes of the execution of representative jobs. Examples of the application of the trace data are: (1) op-code frequency counts, (2) costs of different supervisor functions, (3) usage of supervisor functions, and (4) distribution of execution cycles among supervisor and various user tasks.

A Technical Report, entitled, "The Design and Implementation of an Operating System Tracer," is being prepared for publication, It describes the aspects of the tracer which make it unique, provides some examples of its use, and points the way toward further application of both the data and the technique.

2. Memory Interleaving
(F. W. Terman)

A model of interleaved memory systems for IBM System 360 and System 370 architecture has been investigated by means of a trace driven simulation. The model used is an extension of one developed by G. J. Burnett and E. G. Coffman, Jr.[1] The trace data to drive the

-----

1. Burnett, G. J. and E. G. Coffman, Jr., "A Study of Interleaved Memory Systems," Proc. AFIPS Spring Joint Computer Conf., vol. 36, AFIPS Press, Montvale, New Jersey, 467-474, 1970.

simulation was obtained from the trace facility described above. The simulation determined the effect on the memory bandwidth of the degree of interleaving, the length of the word transferred, etc.

The theoretical predictions of the Burnett-Coffman model were found to fit well with the simulation results for the fetching of instructions. For the fetching of operands, however, the simulation results show only about half the increase in the memory bandwidth that the Burnett-Coffman analysis predicts.

A preliminary report on this work was given at the ACM Computer Science Conference in Anaheim, California, on 11 February 1976 and was published in the Proceedings of that conference. A full report of the results to date has been accepted for presentation at the Symposium on the Simulation of Computer Systems in Boulder, Colorado, on 10 August 1976. This report will be published in the Proceedings of that conference.

B. RELIABLE COMPUTERS

E. J. McCluskey

NSF Grant MCS 76-05327

1. Probabilistic Models of Logic Circuits

(K. Parker, E. J. McCluskey)

A model has been developed where the probability that a "1" appears on the output (output probability) of a combinational circuit can be calculated given the input probabilities. These output probability polynomials can be generated by either of two algorithms. One involves Boolean analysis of the logic function, summing component probabilities contributed by the fundamental products of the function. The other proceeds directly from the circuit description, gate by gate, producing the output probabilities of all gate outputs in the realization of the function. This algorithm allows large circuits (up to 20 inputs) to be analyzed by hand with pencil and paper, and in a simple manner, eliminates the problems associated with dependencies among gate input probabilities within the circuit. This algorithm can be implemented using existing symbolic mathematics programs.

With the ability to derive output probability polynomials, one can modify a given circuit to model some faulty behavior and then find the probability of detecting this faulty behavior as another probability polynomial. One can then study factors that may maximize or minimize the probability of detecting the faulty behavior.

For example, circuit synthesis procedures could include such an analysis in order to improve testability (maximization), or examine the degree of fault secureness of self-checking networks (minimization). Concepts that follow from detection probabilities are those of signal reliability and error latency.

These concepts are useful in analyzing methods of random testing and random test generation. For example, the new concept of compact testing can be analyzed. The compact testing approach utilizes random inputs to a circuit under test. Decisions on whether the circuit is operating correctly are made by observing (for example) the number of ones the circuit produces. In the area of test generation, one can study the effects of controlling the statistical nature of a random input source on test generation efficiency. It then becomes evident that the random test generation process can be adaptively controlled with good results.

2. Probabilistic Analysis of Logic Functions (K. Parker, E. J. McCluskey)

Output probability polynomials contain much information about logic functions. Certain types of information are readily obtained by simple evaluation of the polynomial. For example, let function F of n inputs, $Xl,....,X_r$ have an output probability polynomial, $p(x_1,....,x_n)$. Then the number of fundamental products in F is $2^n \cdot p(\frac{1}{2},\frac{1}{2},...,\frac{1}{2})$. There exist sets of x_i such that $p(x_i)$ is unique for all 2^{2^n} possible functions. There exist other sets of x_i that produce other partitions on the set of possible functions.

An implication of this is that a circuit under test could be made to "identify" itself to within an equivalence class, merely by driving its inputs with probabilities x_i and counting the number of l's produced. Current research is examining various such strategies as random testing schemes.

Of further interest is the possibility of doing circuit synthesis and analysis entirely in the probability domain. For example, by calculating output probabilities and retaining the inter-signal dependency information (not dropping algebraic exponents) one may discern the existence of static hazards in the network.

3. Error Latency in Digital Circuits
(J. J. Shedletsky, E. J. McCluskey)

a. The Error Latency of a Combinational Digital Circuit

In a digital circuit, there is typically a delay between the occurrence of a fault and the first error in the output. This delay is the error latency of the fault. A method to probabilistically predict the error latency of a fault is given. A simple technique establishes a bound on the error latency of a fault for a given circuit.

The error latency behavior of a circuit susceptible to one of many possible faults must be described by an "average" of the latent effects of each fault. This "averaged" measure is the circuit error latency. Methods to approximate the circuit error latency are given.

It is shown for a very general case that the circuit error latency is relatively small compared to a typical functional lifetime of the circuit. The implications of this for life expectancy calculations were explored.

Random testing and random test-set generation are analyzed using the circuit error latency model. It is shown that random testing can be quite effective in testing combinational circuits, yet the analysis required to verify this effectiveness for a given circuit is computationally prohibitive. Random testing is a useful method of testing only if the test confidence need not be accurately verified.

[illegible]. The Error Latency of a Fault in a Sequential Digital Circuit
(J. J. Shedletsky, E. J. McCluskey)

In digital circuits there is typically a delay between the occurrence of a fault and the first error in the output. This delay is the error latency of the fault. A model to characterize the error latency of a fault in a sequential circuit has been developed.

Random testing of sequential circuits has been analyzed using the error-latency model (ELM). For a desired quality of test, the necessary length of the random test may be specified. Conversely, the quality of a test with a fixed length may be calculated. The accuracy of a previous analysis of random testing is shown to be quite poor in some cases.

b. A Rollback Interval for Networks with an Imperfect Self-Checking Property
(J. J. Shedletsky)

Dynamic self-checking is a technique used in computers to detect a fault quickly before extensive data contamination caused by the fault can occur. When the self-checking properties of the computer circuits are not perfect, as in the case with self-testing only and partially self-checking circuits, the recovery procedure may be required to roll back program execution to a point prior to the first undetected data error caused by the detected fault.

A method by which the rollback distance required to achieve a given probability of successful data restoration may be calculated has been developed. To facilite this method, operational interpretations are given to familiar network properties such as the self-testing, secureness, and self-checking properties.

An arithmetic and logic unit with imperfect self-checking capability has been analyzed to determine the minimum required rollback. distance for the recovery procedure.

4. Performance Evaluation Techniques for Highly Reliable Systems
(J. Losq)

Research for the past year has been centered on performance evaluation techniques for highly reliable digital systems which use redundancy to enhance the reliability.

Basic theoretical research has been concerned with the effects of the unreliability of the mechanisms for fault-detection, replacement and reconfiguration. It was shown that the best performance is obtained, in general, when systems are provided with only one or two spares[1].

The study of the replacement mechanisms has led to the exploration of a new redundancy technique: self-purging redundancy[2]. This technique is simpler than hybrid redundancy, and the switching mechanisms used are extremely simple and reliable. Very detailed reliability computations have shown that self-purging systems are well suited for a large range of applications.

The performance of redundant systems intended for critical applications depends heavily on the characteristics of the failures. A reliability model has been developed that takes into account the various degrees at which failures affect the overall systems[3]. Failures limited to one copy can invalidate it either completely or only partially. Multiple failures, that affect several copies, are also taken into account. They are especially dangerous because the redundant systems that use the majority as the standard for correctness may not be able to recover from these faults. A

---

*1 Losq, J., "Influence of fault-detection and switching mechanisms on the reliability of stand-by systems", Digest of Fifth International Symposium on Fault-Tolerant Computing, Paris, France, June 18-20, 1975, pp 81-86.

2 Losq, J., "A Highly efficient redundancy scheme: Self-purging redundancy", IEEE Trans. on Computers, Vol. C-25, June, 1976, pp. 569-577.

3 Losq, J., "Multiple failures and redundant systems", Proc. 1976 Conf. on Information Sciences and Systems, The Johns Hopkins University, Baltimore, MD, March 13, April 1, 2, 1976 pp. 384-389.

strategy that guarantees maximal recovery from multiple failures has been developed.

Recently, special attention has been devoted to the problem of reliable multiprocessors. These systems use the technique known as graceful degradation which allow them to survive failures by reducing the quality of the service they provide. A general model for the evaluation of their performance is being developed. The model takes into account hardware as well as software failures, and the various types of failures and their different effects are also considered. It should provide a systematic way to evaluate overall system performance (i.e. reliability, safety, processing power, etc) as well as the reliability of the results of programs run on any kind of gracefully degradable multiprocessor. A report should be available in the near future.

Besides the study of reliability, research efforts have been devoted to the problem of random testing. A report has been written[4] on the evaluation of the efficiency of a random testing method that does not require the use of a "gold unit" but rather uses some kind of signature. This method trades test length for avoiding complex analysis of the circuits (or gold units). It was shown that the probability that faulty circuits pass the test decreases as the inverse of the square root of the test length. Up to now, the evaluation has been restricted to combinational circuits, but it is hopeful that sequential circuits should soon be similarly studied.

[4] Losq, J. "Reference-less random testing", Digest of Sixth Int'l Symp. on Fault-Tolerant Computing, Pittsburgh, Pa., June 21-23, 1976 pp. 108-113.

5. Procedures to Obtain Optimal Test Sequences
(A. Verdillon)

Procedures that give optimal sequences for both detection and location of faults in sequential circuits have been investigated. Simulation procedures have been studied that use a very compact representation of the behavior of normal and faulty machines: the test flow table.

For the detection problem theorems show how d-dominance on the set of faults and on the set of sequences reduces strongly the set of faults and the set of sequences to consider. A detection graph whose arcs correspond to sequences of inputs is obtained and leads easily to optimal test sequences.

The general approach to extend detection procedures for fault location is to consider not only all the products of the fault-free machines by every faulty machine, but also all the products of faulty machines two by two. With the concept of location expression, the construction of the product of $\binom{m}{2}$ machines is not required (m is the number of faults). The location expression allows us to define l-dominance on the set of faults and on the set of sequences. The simple extension of theorems for detection to the case of location gives the location graph from which optimal diagnosis sequences are obtained.

Finally, we have shown how analogous procedures can be used for asynchronous circuits and for faults that affect the reset circuitry.

6. Reconfigurable Computer Systems
(M. L. Blount)

This research is directed toward the development of a general purpose, fail-softly, high availability computer system. Reconfigurable modules are essential for the graceful-degradation property of computer systems. Processing elements, memory modules, channels, control units and device controllers are the reconfigurable modules that will be employed in the design. A pure symmetric multi-processor with a multi-module main memory has been chosen as the nucleus architecture. A model has been developed that will give an approximation to optimal main memory size that minimizes the (memory) word-hours lost due to covered and uncovered faults. Current investigations into the architecture specification are directed toward determining the minimal number of redundant connections that should be placed between channels and control units in order to minimize I/O device-hours lost due to covered and uncovered faults.

Efforts have also been directed toward determining how faults cause fail-softly systems to crash. The set of all faults can be divided into the classes of covered and uncovered faults. Covered faults are those for which detection, diagnosis, reconfiguration and error recovery are all successful. The remaining faults are uncovered faults. Faults which cause a fail-softly system to crash are: (1) uncovered faults, (2) covered faults that result in a nonoperational configuration, and (3) faults that occur during the recovery process.

We have conjectured that uncovered faults are the major contributor to system unavailability. As a result, we have chosen to direct a major research effort at determining effecient diagnosis, reconfiguration, and error recovery strategies for fail-softly systems.

There are two main models for diagnosis; they are the Preparata-Metze-Chien model and the Kime-Russell model. We are developing a probabilistic characterization of the Preparata-Metze-Chien model. This will allow us to characterize incomplete test coverage and a self-diagnosing module. Both of the above models lack this ability.

7. Self-Checking Circuits for Cyclic Redundancy Codes (D. J. Lu)

The applications of cyclic redundancy codes in computer storage systems, digital communications equipment, and built-in testing logic have been surveyed. The encoders and decoders for cyclic redundancy codes are usually implemented as linear feedback shift register circuits, Typical circuits (including a commercially available LSI device) for error detection by cyclic redundancy codes have been studied. More complicated circuits and cyclic redundancy codes for error correction have also been examined. Present methods for detecting faults in the coding circuits include duplicated circuits, parity prediction, and periodic testing. A study of failure modes in the coding circuits is being performed as part of an effort to reduce the hardware and time overheads presently required when providing the coding circuits with self-checking capabilities,

8. Design of Fault-Tolerant Iterative Cell Arrays (R. C. Ogus)

An iterative cell array consists of a cascade of identical combinational circuits, called cells. In unilateral arrays, signals propagate through the array from left to right only. Iterative cell arrays have important applications in digital systems. Recently an iterative cell array was used in an effective way to implement the switch in a hybrid redundant switching scheme. In the Siewiorek-McCluskey switch design, the iterative cell array determines the actual switching of the spare modules.

A design method has been developed to increase the fault-tolerance of iterative cell arrays. The sets of inputs and outputs of each cell are encoded by means of error-correcting codes, and the cell table is modified to map error input states to the appropriate correct output states. The fault-tolerant iterative cell array has been used in a fault-tolerant switch design for hybrid redundancy, and was shown to improve the system reliability.

Work has also produced a procedure for minimizing the logic in the coded cells. Design of the coded cells becomes unwieldy using conventional Boolean minimization techniques due to the large number of variables involved, A design procedure has been determined to minimize the cell logic based on the concepts of distance and intersection. The procedure is an extension of a method suggested by Russo for counters. The extension of Russo's method to handle iterative cell arrays has been described, together with a discussion of the minimality of the circuits produced by the design procedure. Some examples of one-fault-tolerant circuits have been characterized, and the extension of the procedure to handle general t-fault-tolerant arrays has been carried out. The description of this work has been reported in a Technical Note.[1]

9. Distributed Computer System Modelling
(M. D. Beaudry)

Reliability measures for distributed or multiprocessor systems should include some consideration of total performance. Traditional reliability calculations are applicable when a high level of redundancy is used to achieve fault tolerance, but these measures do not consider the problems of diminished or degraded performance. A more useful system characteristic for distributed computer systems would be the effective reliability, i.e., the reliability as a function of available computing time rather than "wall clock" time. Mathematical models for such multiprocessor systems are being developed to study effective system reliability and system availability.

- - - - -

1. Ogus, R. C., "Design of Fault-Tolerant Iterative Cell Arrays," Tech. Note no. 85, Digital Systems Laboratory, Stanford University, Stanford, California, December 1975.

10. Dual Redundant Computer Systems
(M. D. Beaudry)

Models and applications of dual redundancy which are relevant to computer systems were examined. The models studied were of various dual configurations and included systems with and without repair. System parameters of interest were:

(1) the distribution of the time before failure

(2) the mean time before failure (MTBF), and

(3) the system availability, i.e., the probability that the system is functioning correctly.

These system measures were related not only to the mean time to repair (MTTR), but also to the distribution of the repair times. Markov models were extensively used and the resulting differential equations solved by well-known techniques, such as Laplace transforms. The systems treated included such considerations as non-instantaneous switchover of the spare unit and periodic checking.

11. Fault Detection in Combinational Networks
(J. Savir)

The area of multivalued logic has been studied with an emphasis on realizations of multivalued combinational networks and minimization of n-valued Boolean functions with multivalued gates.

In addition, prior work on fault detection in modular combinational networks (performed by J. Savir) and fault dominance in combinational circuits (performed by K. Mei) were combined as an area of investigation into the category, "properties of faults and fault detection in combinational networks". This work establishes the necessary faults (and, hence, the necessary lines) needed to be tested in a given combinational network in order to guarantee the detection of all single faults in the network. The problem of efficiently testing a network built of reliable modules with unreliable interconnections has been solved. It is shown that in a network built of "unate modules" one needs only to test the input to the network and the reconverging paths in order to test all faults in the intermodular lines. A paper on this combined subject is in preparation.

Currently the question of fault latencies in combinational networks subject to intermittent faults is being investigated. The model used for intermittent faults considers the faults to be active for a small fixed period, δ, and to be inactive for a period, τ where τ is assumed to be a random variable.

C. EVALUATION OF DUAL-COMPUTER CONFIGURATIONS

E. J. McCluskey and R. C. Ogus

NASA Grant 05-020-699, Sup. 1

For the past few years, work has been underway under the sponsorship of NASA Ames Research Center to study reliable computer systems for use in Short Takeoff and Landing (STOL) aircraft. A prototype system has been developed by the Charles Stark Draper Laboratories (CSDL), Cambridge, Massachusetts, for this purpose, and was delivered to the NASA Ames Research Center in mid-1975 for a series of flight tests under the direction of personnel from NASA. This system known as the SIRU system consists of a set of navigational instruments <the Strapdown Inertial Reference Unit (SIRU)>, and a digital computer complex to process information from the SIRU and display this information to the pilot of the aircraft. The SIRU navigation package is fault-tolerant and it was desirable to develop a reliable computer system that would match the high reliability of the SIRU instrument system. The computer complex used is a dual processor, real-time system using two Honeywell H316 central processing units, with a special purpose arbiter component to evaluate the operation of the two Honeywell processors. Both processors execute the same algorithms, and the arbiter designates one processor as master and one as backup for each basic operational cycle.

The initial computer system design was studied by the Center for Reliable Computing (CRC) at the Digital Systems Laboratory (DSL), and the design of the final system evolved with discussions and collaboration between CRC and CSDL. Meetings were held at CSDL in November 1973, May 1974 and December 1974 to review the progress of the design. CSDL carried out the actual construction of the system.

Research work during the past year has concentrated on the reliability evaluation of the dual computer system configuration. The evaluation consists of three basic approaches, viz., analytical modelling, simulation and experimentation. It is felt that the combination of the three separate approaches will provide a comprehensive tool for the reliability evaluation of digital systems. A general purpose simulator is being developed at CRC which can handle a wide variety of digital systems. In addition, a laboratory facility is being constructed which will enable actual physical experimentation on prototype systems. The actual prototype of the SIRU dual computer system has been made available to CRC for the study, and the parallel simulation and experimentation approach is being carried out in addition to the analytical reliability modelling efforts.

1. <u>The CRC Simulator</u>
(P. A. Thompson, R. C. Ogus)

The use of simulation can be a very powerful technique to evaluate the reliability of computer systems. Analytical reliability modelling usually requires numerous simplifying assumptions to be made to render the mathematics tractable. Simulation allows many of these assumptions to be removed, resulting in a more accurate model of the system behavior. Injection of faults into the system is more flexible and a variety of statistical measures can be computed to characterize the system reliability.

A very general purpose simulator is currently being developed at the CRC. The basic simulator is general enough to allow practically any type of hardware configuration to be simulated. The actual system configuration is specified by the user, as is the level of detail. The system configuration is built up by the user by supplying subroutines which characterize each unit in the system. The user can also study a portion of the system in detail and then incorporate the results in a higher level simulation of the total system, where the subsystem studied in detail is represented as a single unit. In this way, the simulation time can be reduced.

The simulator model considers the hardware system to be composed of a set of independent units, connected by data links. The user specifies the type of each unit and the behavior of each type of unit. Each unit has its own inputs which are link data values, current unit states, fault probabilities, time delays between input and output links, and other special parameters depending on the type of unit. The unit outputs are link data values, current unit states, and fault data. The user specifies the behavior of each unit type in terms of the relation between its inputs and outputs, then builds the complete system by assigning a type and other parameters to each unit, and connecting units with other links.

Simulation of the system in operation consists of processing each unit (recomputing the unit's outputs) whenever one or more of its inputs change. Future changes of all outputs are stored in a list, called an event list, which is ordered according to the time of each change. The simulated current time always jumps to the time change of the next system event. The simulation is thus event driven.

Each unit may represent a simple component like a logic gate, or a complex module such as a processor. The link data values may be logical l's or O's, state probability vectors, navigation data, or a number signifying the correctness of the data on that link in the real system. It is thus possible to determine the effects of faults in detail on a subsystem, and then to incorporate the results into a single module as one unit of a larger subsystem. The level of detail for the entire system is not limited by the program, and by merging analyzed subsystems into single units, the computer time for a detailed simulation is reduced. It should be emphasized that building up the system to be simulated does not involve changing the simulation software. Defining the behavior of a unit type requires writing a short subroutine which computes the outputs from the given inputs, and the rest of the system is a table of unit parameters. The random event generators are supplied by the basic software package, which can cycle through many missions of a system configuration in a Monte Carlo simulation, The simulation package is thus very general in terms of the types of system configurations it can handle, as well as in terms of the level of detail. Changing the structure of the simulated system does not involve rewriting simulation software, but rather requires just the addition of new subroutines to specify the different units.

Initial use of the simulator has shown that it is an effective tool to compute reliability parameters. The simulation results correspond very closely to analytically computed curves, and can be improved even further by running the simulator for a larger number of missions. It has also been apparent that many simplifying assumptions, which are necessary in analytical models to make the mathematics tractable, can be very easily removed; all that is necessary is the changing of some input parameters to the simulator. In this way, accurate representations of the system can be obtained which would be difficult to handle mathematically.

The simulator thus allows a wide variety of system configurations to be modeled for a wide range of component failure specifications. Although the immediate use of the simulation package has been the study of the dual computer system, the software is general enough to allow major changes in the simulated system configuration, so that various failure recovery techniques and hardware configurations can be compared in terms of how they affect the overall system reliability.

2. Experimental Test Facility
(A. F. Hunter, R. C. Ogus)

A laboratory test facility is being constructed at CRC which will enable physical experimentation on prototype systems. The initial experimentation will be performed on the NASA dual computer system. In addition, the construction of a self-checking processor is being planned at CRC and experimentation will also be carried out on the completed prototype.

The description of the current experimentation effort can best be described with reference to the dual system. However, it is hoped that the initial effort will produce guidelines for experimentation on more general computer systems.

The prime interest on the dual computer system is the redundancy management, both hardware and software, and thus the actual navigation instruments were disconnected from the system.

The software has two levels of operation, viz., the navigation calculations and a background of self-test programs. It was decided to interface a third minicomputer to the dual computer complex which would take the place of the instruments. However, since the navigation calculations are not of prime interest, much of the navigation software could be dispensed with and replaced with much simpler foreground software.

Hardware has also been designed and constructed to interface the third minicomputer to the dual complex. This interface allows the third machine to interrupt either or both of the dual processors and make modifications to the processor's memory, In this way faults can be selectively injected into the system.

Thus, the dual system will be operated and faults randomly injected into the processors. The outputs of the dual system are recorded on magnetic tape for post-experimentation analysis. Currently active, are studies of the fault-injection mechanism and the gathering of output data, from which reliability, fault-tolerance and other measures can be calculated.

Other facilities available in the experimental test facility include software translators on a larger (IBM 370) machine, as well as automatic down-line loading of object modules from the 370 to the laboratory via telephone lines.

3. Analytical Modelling
(P. A. Thompson, R. C. Ogus)

By using analytic models of typical NMR computer systems with voters, it is possible to study the effects of voter faults on the overall system reliability. The failure rates of digital systems are related to their physical and electronic complexity, and the relative complexities of the voters and redundant computers is therefore an important factor to consider when designing an NMR system. We have found optimal boundaries for the complexity ratio to be functions of the degree of redundancy, the desired improvement of mission time for the system, and the system reliability desired at the end of the mission. The complexity of the voter is also related to the number of redundant computer modules. We have produced curves describing the tradeoff between the mission time improvement due to increased computer redundancy and the degradation resulting from the implied increase of voter complexity. The analysis clearly shows that the reliability of an NMR system depends greatly on the relative complexities of the voter and redundant modules, and equations have been found which may be used to achieve the optimal redundant configuration for a given situation,

A dual computer system with an imperfect arbiter has also been studied analytically. The decision function of an arbiter is unlike that of a voter because its choice is based on various error-detection techniques which do not guarantee to detect <u>all</u> errors. The ability of a non-faulty arbiter to make a correct decision is in general related to the arbiter complexity, so there is a tradeoff in reliability due to a more "intelligent" arbiter having a higher failure rate. Curves have been produced which define the optimal arbiter complexity for increased reliability, given the probability of making a correct choice as a function of circuit complexity. The probability that the arbiter makes a correct choice when one of the two computers is faulty can have a significant effect on the overall system reliability.

It is possible that making one computer more complex than the other (e.g., using different amounts of error detection) could increase system reliability by changing the probability of making a correct choice when one is faulty. A set of sufficient conditions for this have been derived, relating the choice probabilities to the voter and computer failure rates. We have proved the existence of one class of computers for which a balanced dual system is optimal, and are investigating the possibility of another class for which the unbalanced system yields a greater reliability,

D. LSI MULTIPROCESSORS

J. F. Wakerly and M. J. Flynn

JSEP Contract N00014-75-C-0601

1. Design of a Self-Checking Minicomputer Based on LSI (J, F. Wakerly, R. T, Cutler)

A self-checking computer has redundant hardware that enables dynamic detection and diagnosis of hardware failures. For some time we have been studying design techniques for self-checking processors. We are presently enhancing these techniques so that low-cost self-checking computers can be designed using commercially available LSI bit-slice components. Our studies indicate that it is possible to guarantee the detection of arbitrary failures affecting a single LSI package with somewhat less than 50% overall redundancy.

The design and construction of a 16-bit minicomputer is currently underway. Dubbed SCAMP (Self-Checking and Maintainable Processor), the machine uses a 4-bit residue check to detect errors in registers, memory, and arithmetic operations, a 4-bit interleaved parity check for the microprogram memory, duplicated microprogram sequencers, and a fail-safe system clock checker. The 4-bit data path slices for the machine have been constructed and tested, and work on the microprogram sequencer and microcoding is currently being done. The target machine architecture has all of the features of a typical minicomputer and will execute a 16-bit register-to-register operation in about 1 microsecond, including error detection activity.

2. Microprocessor Reliability Improvement Using Triplication
(J. F. Wakerly)

The use of Triple Modular Redundancy (TMR) to improve microprocessor reliability has been explored. It has been shown that the mission time of a small system can be improved by a factor of three or more using this technique. This work has been completed.

3. Synchronization of Multiple Independent Processors
(D. Davies, J. F. Wakerly)

This study investigates the issues of reliably synchronizing multiple processors, with the eventual objective of implementing an N-Modular Redundant (NMR) system using commerically available machines. A theoretical approach has been taken in specifying a system of machines and synchronizers and proving this system will function correctly in the presence of any f arbitrary or malicious faults. As a first practical example, a Triple Modular Redundant (TMR) version of the system has been built with TTL which produces three synchronous square waves and could be used as a fault-tolerant clock with a maximum frequency of 4 MHz. The clock performed correctly with any chip or wire removed and a pulse generator substituted in its place.

In addition, a simple method has been found to initially synchronize the NMR system without using any common reset or synchronization lines. An upper bound has been found for the time required by this procedure.

Finally, resynchronization after transient failures has been investigated. A set of criteria specifying the amount and rate of information transfer necessary for a machine to recover in a given time has been proposed.

Future studies will extend our work on the reliable synchronizer to more complex examples while investigating resynchronization issues, such as tradeoffs of time vs. interconnectivity and methods of achieving the necessary connectivity between modules without sacrificing modular independence.

4. MINERVA - A Multimicroprocessor Network
(L. C. Widdoes)

The Minerva Network is a small experimental microprocessor network based on commercially available microprocessors. It consists of a compatible set of devices organized around a single demand-multiplexed bus. The bus supports transfers of 32-bit data at a maximum rate of 2 MHZ (64 Mb/sec). Devices on the bus include various peripherals, public memory, and up to 28 processors. Currently four 8-bit microprocessors are used. Plans call for the inclusion of several 32-bit processors. The microprocessors have local storage and an instruction cache to minimize the bus interference created by access to public memory. The network has been constructed and is currently in the debugging phase.

The Minerva Network is intended to provide a vehicle for experimentation with both hardware and software aspects of distributed computing. Among the hardware questions which we have investigated are the following: What simple mechanisms can be used to reduce the problem of limited bus bandwidths in a multiple-microprocessor system? How cost effective are multiple-microprocessor systems? What bus structure and arbitration disciplines are suitable for such systems?

The next goal of the Minerva Network project is to investigate the problems of programming the network. We intend to write an operating system which will allow us to program the network as a single entity without giving explicit consideration to the number of processors available.

5. Functional Completeness in Multi-Valued Logic Systems
(R. B. Lee)

Radices higher than binary may be used to reduce wiring complexity and pincount restrictions in current LSI technology. The number of different functions of k variables in an R-valued logic system is R^{R^k}, an exponentially growing set. We are therefore interested in the minimum set of functions that is functionally complete. This relates directly to the number of different types of gates that have to be fabricated for components with higher-than-binary radices.

It is well known that for $R = 2$, the size of a minimal functionally complete set is 1 (a NAND or a NOR function). This study has found that for any $R > 2$, the size of a minimal functionally complete set is only 2. It also identifies possible combinations of two operators for a minimal complete set.

6. Efficiency of Overhead Instructions
(R. B. Lee)

The design of new instruction sets and processor architectures should reflect a clearer understanding of the role of "overhead instructions", like program sequence control and memory reference instructions, in the overall efficiency of the machine organization.

As a first step in this direction, a study of program sequence control instructions was made and a model of the program control structure was proposed. This led to the proposal of the family of general prefetch algorithms to accurately anticipate instruction references, in contrast to the common demand fetch policy, which is at best naive in the context of multi-leveled memory hierarchies.

7. Bounds for Maximal Parallelism
(R. B. Lee)

This is a theoretical study of the performance limits of a single program running on a large array of multiprocessors. It is clear that multiprocessor efficiency is limited by program behavior--especially the control structure as measured by the "fork potential".

The performance of P multiprocessors is bounded by the program entropy or procedural entropy which is related to the logarithm (base the "fork potential" of the instruction set) of the number of outcomes of a given invariant program.

It has been found that the maximum parallelism (or the overlapping of execution beyond which little speed improvement may be observed) is related to 2 raised to the power of the program entropy.

E. COMPILER IMPLEMENTATION

T. H. Bredt and M. T. Kaufman

NASA Grant NCA2 OR745 607

Previous work in this area* has resulted in the specification of a programming language oriented towards the area of fluid dynamics simulations. In this language, vector and mesh operations such as differencing and functional evaluation at each point are available as primitive operations. The intent is to allow the compiler to abstract those operations which may be performed in a highly parallel manner on computers such as the CDC 7600.

Current studies are to carry the implementation forward. Table driven lexical and syntactic analyzers have been designed and are being implemented. Though overriding demands on the time of the principals at NASA have resulted in less progress than had been hoped for, the project is continuing and is still of practical importance and interest to fluid dynamicists.

*A Compiler for Fluid Dynamics, T. H. Bredt, D. L. Hanson.

F. IMPLEMENTATION OF RELIABLE SOFTWARE SYSTEMS

T. H. Bredt

AFOSR Contract F44620-75-C-0082

1. Design and Verification of Real-Time Systems
(J. V. Phillips, T. H. Bredt)

A methodology for the design and verification of a class of real-time systems frequently encountered in applications of digital control has been investigated. These methodologies are described in the form of a design and informal verification of a system used for navigation control of airborne vehicles. The high reliability standards required for such tasks suggest the use of fault-tolerant hardware and software.

A design in being studied for both an abstract version of the system running in an ideal environment and for a version running in a non-ideal fault-prone one. A specification language for real-time systems is being developed as an extension of the programming language PASCAL that provides for concise and clear system descriptions while preserving the degree of efficiency and portability needed to qualify as a practical high-level system implementation language.

An informal verification of both designs is being developed.

2. A Verified Specification of a Hierarchical Operating System (A. R. Saxena)

The design, specification, and verification of computer operating systems is being studied. The operating system problem being considered, that of the many-process problem, is the design of an operating system that can support a large number of concurrent processes. This design problem is a vehicle to investigate: (1) the use of a design methodology (the hierarchical-levels-of-abstraction methodology); (2) the use of structured programming techniques in the specification of the system; and, (3) the development of techniques for the verification of concurrent programs, particularly operating system programs.

A solution to the many-process problem has been obtained and shows that the hierarchical-levels-of-abstraction methodology simplifies the conception of the solution and helps avoid potential deadlocks in the system. A PASCAL specification of the four levels of the system demonstrates the usefulness of structured programming techniques for specifying operating system programs.

A detailed description of the development of the simple memory manager, a complex and large segment of the system, has been formulated to show the use of step-wise refinement for improving the efficiency of the program and as an aid in understanding its final specification, The specifications for the first two levels, simple scheduler and simple memory manager, have been formally verified. The notion of exclusive access of a resource has been formalized and used in the verification of concurrent programs. Sufficient conditions for verifying the absence of deadlocks in a system of monitors have also been developed.

G. PARALLEL COMPUTER SYSTEMS

T. H. Bredt

NSF Grant GJ-41644

State Restoration among Communicating Processes
(D. L. Russell)

In systems of asynchronous processes using messagelists with SEND-RECEIVE primitives for inter-process communication, recovery primitives have been defined to perform state restoration: MARK saves a particular point in the execution of the program; RESTORE resets the system state to an earlier point (saved by MARK); and PURGE discards redundant information when it is no longer needed for possible state restoration.

Errors may be propagated through the system, requiring state restoration also to be propagated. Different types of propagation of state restoration have been identified. Data structures and procedures have been proposed that implement the recovery primitives.

In ill-structured systems the domino effect can occur, resulting in a catastrophic avalanche of backup activity and causing many messagelist operations to be undone. Sufficient conditions have been developed for a system to be domino-free. Explicit bounds on the amount of unnecessary restoration have been determined for certain classes of systems, including producer-consumer systems, k-producer systems (cyclic systems of k producers and k messagelists), and MRS systems (where the sequence of recovery primitives has been described by the regular expression (MARK; RECEIVE*; SEND*)*).

H. ORGANIZATION OF COMPUTER SYSTEMS

M. J. Flynn

ERDA Contract AT (04) 326 PA 39

This project is directed at general studies in the organization of computing machines, including parallel processors which attempt to simultaneously manage a large number of data, logical resources and/or tasks.

Detailed investigations of computer systems include:

(a) Computer system resource use and specification by general models and simulation;

(b) Comparative study and evaluation of system design architectures and concepts, including parallel processors and microprogrammed processors.

1. An Emulation Research Laboratory
C. J. Neuhauser)

The primary mission of the Stanford Emulation Laboratory is to study the processor instruction execution as it is represented by both physical and conceptual processors.

A facility has been developed which may stand in the place of (or emulate) the instruction processing activity of an arbitrary machine. At the center of this facility is an interpretively programmed host machine. When properly programmed, this machine exactly emulates the instruction fetching, decoding and execution of a selected target machine (conceptual or physical). The software oriented process of organizing the host machine resources for a particular target machine emulation is referred to as microprogrammed interpretation.

The Emulation Laboratory is divided into two cooperating but independent subsystems: the emulation system and the experiment control system.

The emulation system consists of a special purpose "host" (or interpretive) processor, control storage, main storage and a common bus system. To emulate a particular target machine, a micro-program interpretation of this machine is constructed and loaded into the control store of the host processor. During the emulation experi-ment, the control store and the physical resources of the host processor serve as the data storage and manipulation resources of the target machine. The main memory system is similarly configured to represent the main memory system of the target machine.

The experiment interface system consists of low speed, user oriented I/O devices clustered around a general purpose processor. The two laboratory systems are coupled via the bus system and are organized so that the experiment control system has direct access to the data storage of the host processor. This provides a convenient method of direct observation and control of the emulation experiment.

Host machine microprograms are written in one of two specially developed languages.

An emulator capable of processing the complete IBM 360 basic instruction set has been written, For typical instruction sequences about 100,000 target machine instructions are executed per second. This means that the emulator is operating at about 70% of an actual IBM 360 model 50 machine. Other performance estimates indicate that comparable execution rates may be expected for similar second and third generation physical processors. Studies are underway to extend the 360 emulator so that detailed resource usage statistics may be gathered during the emulation process.

2. Parallel Processor Studies
(P. Yu)

Our efforts in analytic models of performance for a certain type of parallel structure, which we call SIMD, were not so successful. We have developed in the past the now well-known "log conjecture": that in certain types of parallel processor structures the addition of more processors will not linearally aid in the execution of programs but rather will be only effective as the log of the number of processors. Several years ago we were successful in developing a "splitting" explanation for this phenomenon. However, this was largely conjectural. This year a good deal of effort was spent in using computational complexity models to achieve the same type of log bounds on certain generalized classes of networks. So far these attempts have been unfruitful; the bounds we arrived at have proved trivial and not really indicative of the log observation.

3. Memory Hierarchies and Program Behavior
(B. R. Rau)

The effectiveness of a memory hierarchy typically depends on the tendency of programs to concentrate their references to a subset of their address space over significant periods of time. If this subset can be contained in one of the higher levels, the processor will be able to access information at the speed of the higher level for most of the time, and will encounter longer access times only when the subset changes.

The performance of the memory hierarchy (measured by the average access time that the processor sees) clearly hinges on the currently active subsets being accurately identified. This is the task of the memory management algorithm whose goal is "to have present in the fastest level the information that the processor needs." In general, this entails predicting the future needs of the processor and moving that information up to the higher levels, Since the capacity of each ievel is limited, this will require that some information be displaced. The memory management algorithm must, therefore, be able to accurately determine whether the information currently present in the level will be used in the near future.

That the memory management algorithm is central in determining the performance of large computing systems has been documented extensively in the literature with accounts of high-speed computers which have hobbled along at a fraction of their capacity due to the use of faulty algorithms.

The emphasis during the first phase of this research was on understanding, at an empirical and intuitive level, the manner in which programs behave. This was necessary to understand the properties of interest when it came to a more formal statistical analysis. It was also of importance in isolating the properties which must be modelled.

The second phase, which has just begun, is concerned with a statistical analysis of program reference strings. Preliminary results show that programs typically behave in a manner far more complex than earlier models would lead one to believe. Though requiring the use of more complex models, this also offers the possibility of developing predictive algorithms for memory management and more sophisticated replacement algorithms. This possibility is being investigated.

The third phase will be to propose and validate a model of program behavior based on the statistical analysis of the second phase. This model will then be used to develop analytic techniques for the performance evaluation of memory hierarchies.

I. FEASIBILITY OF REALTIME EMULATION

M. J. Flynn

AFOSR Contract 75-2865

The primary emphasis of this study is on the feasibility of advanced, more powerful host processors which would be capable of processing 1 to 5 MIPS of target machine instruction.

Feasibility of Real-Time Emulation
(W. A. Wallach)

This project has studied several alternate methods for the realization of high performance emulation. High performance or real-time emulation occurs when a host machine is able to interpret the instructions of another machine (called the image machine) in the same time as that machine would have executed the same set of instructions. Occasionally, such interpretation occurs at an even faster rate than the original image machine. We label this phenomenon hyper-real-time emulation. Several organizations have been studied as well as organizational extensions to our present EMMY organization. The most promising structures that we have developed are extensible, overlapped processors. An independent, order-of-magnitude , performance improvement is available through other techniques called directly executed languages.

J. DIRECTLY EXECUTED LANGUAGES

M. J. Flynn

ARC-D Grant DAAG-29--76-G-0001

This research investigates Directly Executed Languages, which serve both as the output medium for a language translator (compiler) and as the input medium for a language interpreter (emulator).

Directly Executed Languages
(L. W. Hoevel)

Significant reductions in the space and time needed to represent and evaluate computer programs have been achieved by applying a new theory of language synthesis called, "Directly Executed Languages." Programs now may occupy 20 times less space than on traditional third generation computers (e.g., System 360),and execute 10 times faster when both the new and old computer systems are implemented in the same technology.

The new theory addresses the problem of matching hardware resources to the requirements of high-level, user-oriented programming languages. An innovative analysis of:(1) the function and interaction of various classes of machine instructions, (2) the ways in which data is referenced, and (3)the selection of machine operators-- forms the foundation of this theory. Results to date indicate that the theory is not unique to a specific programming language or collection of hardware. Several avenues of further investigation are suggested which will greatly expand our understanding of language use and interpretation.

K. DESIGN AUTOMATION OF DIGITAL SYSTEMS

W. M. vanCleemput

JSEP Contract N00014-75-C-0601

1. Mathematical Models for the Circuit Layout Problem (W. M. vanCleemput)

The problem of laying out printed and integrated circuits shows a striking similarity with that of embedding a graph in the plane. This has led to several attempts to use graph-theoretical methods for solving the circuit layout problem. The first step in this so-called topological layout method consists of building adequate graph models for the circuit in question. The result of this research is a method for constructing graph models for complicated components. These models represent the physical and logical equivalence of terminals and the logical equivalence of subcomponents. An interesting property of these models is that they often allow pin and gate assignment to be done in function of an optimal layout.

2. Planarity Testing of Partially Oriented Graphs (W. M. vanCleemput)

Partially oriented graphs can be used as an abstract model for the circuit layout problem. An important step in the topological approach to solving the circuit layout problem requires the partially oriented graph model to be tested for planarity. An efficient algorithm has been developed for this purpose. This algorithm is based on Tarjan's algorithm for testing the planarity of simple graphs and requires a run time proprotional to the number of vertices.

3. A Language for Describing the Physical Structure of Digital Systems (W. M. vanCleemput)

In the past, a large number of languages has been developed for the description of digital systems. Most of these languages describe only the behavior of a system but not its physical structure. The purpose of this research is to develop a powerful language for describing the physical structure of a digital system. Such a language can be used as the input to a large number of design automation tools such as logic simulators, fault simulation, printed circuit layout systems, automated logic diagram generators etc. In the future, a compiler will be developed in order to test the usefulness and the feasibility of this language.

4. Interconnection Algorithms for Multilayer Boards (B. Ramakrishna Rau, Thomas Bennett)

Interconnection algorithms for multilayer boards have traditionally been broken up into four relatively disjoint steps. These four steps which succeed the placement step are:

1) Wire list determination,
2) Layering,
3) Ordering (for routing),
4) Routing.

The disadvantage of having disjoint steps is that each step attempts to maximize the objective which does not necessarily reflect the overall goals of the algorithm. It also restricts the feedback from a subsequent step which could help in maximizing the global objective which is to route as many wires as possible. Furthermore, if the routing step is non-iterative and non-topological (as is often the case), the ordering imposed can be a crucial factor. However, even the best ordering will not, in general, be sufficient to maximize the number of wires that are successfully routed.

With this in mind, an algorithm has been designed and implemented, which, it is hoped, will overcome these problems. The purpose being to investigate alternate strategies, the program has been written so as to be rather general. The central module, the router, is an iterative version of the Hitchcock algorithm which is topological in nature. The generality is enhanced by certain novel features which permit various options during the routing of a wire. For instance, it is possible to specify a set of wires which may be intersected or, alternatively, may not be intersected. Also incorporated is the concept of spatial priority which determines which of two intersecting wires need be re-routed. This minimizes the sensitivity to the ordering imposed. The program has been written such that the layering and routing can be effected simultaneously, separately or repeatedly in alternation.

Current efforts are directed towards evaluating the effectiveness of the global strategies and the individual features.

L. NETWORK INTERCONNECTION PROTOCOLS

V. G. Cerf

[R. Crane, Y. Dalal, J. Estrin, R. Karp, J. Mathis, D. Rubin]

ARPA Contract #MDA903-76C-0093

The main problem is the interconnection of packet switching computer communication networks. The major research emphasis has been on the design, analysis, implementation and testing of an internetwork host-host protocol which is both efficient [small in size, able to achieve low delay and high bandwidth], and ultra-reliable [generally able to survive catastrophic failures in intermediate networks, gateways, and the like].

At the time of this writing [June 1976], we have implemented two versions of the internetwork transmission control program [TCP], one for use as a general purpose, multiconnection interprocess communication facility and one for use as a single-connection terminal support facility [remote access to a serving host in a distant network].

The general purpose version is written in BCPL and runs under the ELF operating system on a PDP-11/20. We have only been partly successful in our goals with this version; its robustness is excellent, but its bandwidth [about 16-20 kb/sec] is disappointing. We believe that part of the problem is the result of a multiprocess organization which requires context switching for internal interprocess communication among the processes that make up the TCP. The size is roughly 12.5 K words of 16 bit memory.

By comparison, the assembly language coded TCPO [single connection TCP] is about 1.2 K words long, runs on an LSI-11, and is capable of up to 50 kb/sec [when running on a PDP-11/20]. Its robustness qualities have yet to be tested.

Concurrently with our protocol work, we have been cooperating with other DARPA contractors on the Packet Radio Network (PRNET) development. Our LSI-11 can be connected to the PRNET via a radio repeater housed atop the Durand building. Provision has also been made to connect our PDP-11/20 to this radio repeater to allow it to serve as a network host. In support of access to the ARPANET and PRNET, we built a Bell 303 modem replacement to connect our PDP-11/20 to the SUMEX IMP and a standard BBN 1822 interface to connect our LSI-11 to the PRMET. An 1822 interface was also installed to allow connection of our PDP-11/20 to the PRNET.

We have also continued our work on secure internetwork protocols, co-operating with DARPA/IPT and other agencies to determine the appropriate protocol variations needed to make TCP suitable for encrypted communication.

We participated in the writing of a CCITT and IS0 contribution ["Proposal for an International End to End Protocol," Cerf, McKenzie, Scantlebury, and Zimmermann] on standard international host-host protocols through Working Group 6.1 of IFIP* TC-6. The working group voted to submit the proposal to international standards bodies.

* International Federation of Information Processing

M. THE 2000 TERMINAL COMPUTING SYSTEM

V. G. Cerf

JSEP Contract N00014-75-C-0601

A study has been undertaken to investigate the possibilities, requirements, capabilities, limitations, and advantages of a computing system serving a very large number of users.

The 2000 Terminal Computing System
(W. A. Warren)

Computer and communication networks linking potential users over a large geographic area make possible a computing system with an active user population of several thousand. Functional, rather than general-purpose, multiprocessing may make such a system feasible. Such a system could realize a number of economies of scale and specialization and could provide reliability through modularity and redundancy. We have investigated the fundamental architectural requirements, limitations, functional organization, and specialization of these systems, the software requirements in a network environment, and the functional characteristics of the user interface (both hardware and software).

In our investigation of system architectures feasible for use in a 2000 terminal computing system, we have arrived at a distributed network of fully inter-connected microprocessors. The microprocessors will be individually allocated to the functional tasks of: (1) terminal I/O modules, (2) common memory modules, and (3) the general processing modules. Each module will contain local memory, processing power, and also a communication interface, providing via micro-watt radio a complete interconnection network.[1] This interface will handle module-module packet communications protocol using the carrier sense multiple access technique developed for the ALOHA system.[2,3]

One feature of this system will be the fast response time seen by the terminal users. Each terminal connected to the system will have its own dedicated terminal-I/O module attending to line control, formatting, echoing, etc.[4] Another, perhaps more important feature, would be the potential for hardware reliability, implicit when multiple copies of each hardware structure exist. Such a system, with appropriately designed software, would be able to gracefully cope with the complete failure of one or more hardware modules, resulting in a minor degradation of response time as seen by the user. This type of reliability will be obtained by using the same methods and techniques of Pluribus[5]: duplication of hardware, isolation between independent hardware modules, and a software system which performs self-consistency checking and recovery.

- - - - -

1. Okano, R., "Preliminary Design Considerations for a Multi-Microprocessor System," University of Hawaii, 1974.
2. Abramson, N., "The ALOHA System," _AFIPS Proc._, vol. 37, Fall 1970.
3. Kleinrock, L., "Random Access Techniques for Data Transmission over Packet-Switched Radio Channels," _Proc. Nat'l. Computer Conf_., 1975.
4. Heckel, P. C. and B. W. Lampson, "A Terminal Oriented Communication System," (_BCC-500_), September 1975.
5. Ornstein, S. M. et al, "Pluribus - A Reliable Multiprocessor," _Proc. National Computer Conf_., 1975.

We have also investigated the problems of recovering error-free information when a faulty hardware module is detected. If an error is detected at some point during the processing of a task, it may be necessary to amputate the faulty module, re-configure the hardware environment, and re-run that task which failed. The state of the system, including system and user data structures which were caused to become invalid must be restored to a previously consistent state so that the task may be run again.

A system architecture was investigated which would permit systematic recovery after the occurrence of an error and the restoration of erroneous data, Algorithms were developed which will perform state restoration, minimizing both the recovery overhead cost and the total expected task execution time. The model used for these algorithms provides for the recovery of program state, the revoking of erroneous outputs, and the restoration of external data structures. Further areas of research will extend these recovery techniques to handle those problems which occur during the concurrent, asynchronous interactions between the multiple hardware modules and the organization of the operating system needed to provide this fail-soft reliability.

N. PACKET-SWITCHED NETWORK TECHNOLOGY/COST STUDIES

V. G. Cerf

NSF Grant DCR73 07973

1. Distributed Operating Systems
(Y. K. Dalal)

While attempting to isolate an optimum file migration strategy for a distributed file system, we noticed that for spanning tree networks (non-redundant connectivity) the solution was straightforward because the cost of accessing a file in such a network is monotonically increasing as the accessing site moves farther from the site maintaining a particular file. Our interest in spanning trees and, in particular, in minimal spanning trees was awakened by this realization. We also noticed that it is frequently useful to be able to broadcast a request for access to a file to all possible sites that might be maintaining the file, in the event that we did not know its current location. This led us to ask whether redundantly configured networks (such as the ARPANET or the j-connected networks we worked with earlier in this study) could maintain a routing policy for broadcast messages as well as point-to-point messages.

The idea is simple. There are three obvious ways to deliver a particular message to all nodes in a network (we are ruling out networks, which by their nature are automatically broadcast networks, e.g., the ALOHA Network in Hawaii, ring-based networks, the ALOHA satellite networks, the ETHERNET at Xerox PARC <Palo Alto Research Center>, etc.):

1. Send a copy of the message to each node.

   This requires N copies of the message to be sent and is likely to require that more than one copy of the message traverse the same communication link. This is expensive as it utilizes extra bandwidth and takes more time.

2. Put multiple destination names on the message and use the normal routing procedure of the packet network to deliver a minimal number of copies on each communication link.

   The routing table in each packet switch could be used to decide whether destinations A, B, C would all take the same next link and therefore only one copy of the packet need traverse that link. This is a very efficient policy since there might be different routes to each destination from a particular source. Another way of characterizing this solution is that from a particular source, the network has an implicit spanning tree for routing packets to all destinations. The spanning tree route is different for each source.

3. Impose a logical spanning tree on the more over-connected packet network and send a single broadcast message over this spanning tree to all destinations.

   This is simpler than in (2) above, but it may not be as efficient,

During this last period, we have investigated the creation of minimal spanning trees from distributed packet switching networks through a parallel algorithm which can be executed concurrently by all nodes in the network. We have shown' that the parallel algorithm must

- - - -

1. Dalal, Y., "A Distributed Algorithm for Constructing Minimal Spanning Trees in Computer Communication Networks," Tech. Rpt. no. 111, Digital Systems Laboratory, June 1976.

terminate in such a way that a minimal spanning tree is formed (i.e., routing tables are created which create an MST over the network). A minimal spanning tree does not necessarily have minimum diameter. Our algorithm is not designed to produce a minimum diameter tree, and it is still an open problem as to whether any parallel algorithm will produce a minimum diameter MST.

2. Integrated Circuit and Packet Switching (R. C. Crane)

The focus of attention has been on the modelling of combined packet/circuit switching systems. The "slot stealing" nature of the proposed mixed system will only be successful if "empty" slots occur relatively often. The appearance of a particular "idle" character in an already subscribed TDM slot causes the slot to be marked "stolen" if there is data from a packet to be carried in the slot. At the receiving end, the "idle" character is regenerated and inserted in the outgoing TDM stream. The character contained in the stolen slot is used to reassemble the packet.

The statistical success of the mixed system is strongly dependent upon the statistical occurrence of idle characters in the input stream. If the bandwidth "stolen" exceeds the overhead cost of the "stolen bit" in each TDM slot, then the system effectively recovers additional packet mode bandwidth by the encoding of the idle character.

The size of the TDM slot plays an essential role in the success of the plan. A narrow slot size insures more frequent occurrences of the chosen "idle" code, but the overhead cost for the "stolen" marker bit *is* higher since it occurs more often, even if not used.

APPENDIX

Ph.D. DISSERTATIONS

Kaestner, Olaf - "Architectural Considerations for an Expandable Microcomputer System," Dept. of Electrical Engineering, (Prof. Allen M. Peterson)

Kolupaev, Stephen George - "Cutting Planes and Self-Checking Networks," Dept. of Electrical Engineering, (Prof. Edward J. McCluskey)

Mortenson, John A. - "Computer Storage Hierarchy Models and Analysis," Dept. of Electrical Engineering, (Prof. Forest Baskett)

Narasimha, Madihally J. - "Techniques in Digital Signal Processing," Dept. of Electrical Engineering, (Prof. Allen M. Peterson)

Oliveira-Dias, Francisco Jose - "Multiple Fault Analysis in Combinational Logic Circuits," Dept. of Electrical Engineering, (Prof. Edward J. McCluskey)

Parker, Kenneth Paul - "Probabilistic Test Generation," Dept. of Electrical Engineering, (Prof. Edward J. McCluskey)

Rao, Gururaj Seshagiri - "Performance Analysis of Cache Memories," Dept. of Electrical Engineering, (Prof. Forest Baskett)

Reiber, Johan Hendrikus Christiaan - "Real-Time Detection and Data Acquisition System for the Left Ventricular Outline," Dept. of Electrical Engineering, (Prof. Allen M. Peterson)

Russell, David Lewis - "Error Recovery and Process Communication," Dept. of Computer Science, (Prof. Thomas H. Bredt)

Saxena, Ashok Raj - "A Verified Specification of a Hierarchical Operating System," Dept. of Electrical Engineering, (Prof. Thomas H. Bredt)

Shedletsky, John J. - "Error Latency in Digital Circuits," Dept. of Electrical Engineering, (Prof. Edward J. McCluskey)

Sunshine, Carl Alan - "Interprocess Communication Protocols for Computer Networks," Dept. of Computer Science, (Prof. Vinton G. Cerf)

Ph.D. DISSERTATIONS (continued)

Usas, Alan Michael - "Error Management in Digital Computer Input/Output Systems," Dept. of Electrical Engineering, (Prof. Edward J. McCluskey)

Weller, Daniel L. - "Optimal Searching Algorithms for Parallel Pipelined Computers," Dept. of Electrical Engineering, (Prof. Edward S. Davidson)

JOURNAL ARTICLES, BOOKS, BOOK CHAPTERS

vanCleemput, W. M., Computer-Aided Design of Digital Systems: A Bibliography, Computer Science Press, Woodland Hills, California, 1976.

vanCleemput, W. M., Computer-Aided Design of Digital Systems: A Bibliography, vol. 2: 1975-76 Update, Computer Science Press, Woodland Hills, California, September 1976.

Gschwind, H. W. and E. J. McCluskey, Design of Digital Computers, 2nd edition, Springer-Verlag Publishers, New York, 1975.

Flynn, M. J., "Interpretation, Microprogramming and the Control of a Computer," Chapter 10 in Introduction to Computer Architecture, H. Stone, ed., Science Research Associates, Publishers, Palo Alto, California, 1975, pp. 432-473.

Flynn, M. J., "Microprogramming: Another Look at Internal Computer Control," (invited paper), Proc. IEEE, vol. 63, no. 11, 1554-1567, November 1975.

Wakerly, J. F., "Microcomputer Reliability Improvement Using Triple Modular Redundancy," Proc. IEEE, vol. 64, no. 6, 889-895, June 1976.

vanCleemput, W. M., "Comments on 'Theory of Multiplace Graphs"', IEEE Trans. on Circuits and Systems, vol. CAS-23, no. 6, 403-404, June 1976.

Dias, F. J. O., "Truth-Table Verification of an Iterative Logic Array," IEEE Trans. on Computers, vol. C-25, no. 6, 605-613, June 1976.

JOURNAL ARTICLES (continued)

Losq, J., "A Highly Efficient Redundancy Scheme: Self-Purging Redundancy," IEEE Trans. on Computers, vol. C-25, no. 6, 569-578, June 1976.

Shedletsky, J. J. and E. J. McCluskey, "The Error Latency of a Fault in a Sequential Digital Circuit," IEEE Trans. on Computers, vol. C-25, no. 6, 655-659, June 1976.

Turcat, C. and A. Verdillon, "Recursion and Testing of Combinational Circuits," IEEE Trans. on Computers, vol. C-25, no. 6, 652-653, June 1976.

Cerf, V. G. and R. E. Kahn, "A Protocol for Packet Network Intercommunication," reprinted in, Computer Networking, Blanc and Cotton, eds., IEEE Press, Piscataway, New Jersey, 95-106, 1976.

Cerf, V. G. et al, "Proposal for an International End-to-End Protocol," ACM SIGCOMM Computer Communication Review, vol. 6, no, 1, 63-89, January 1976.

Wakerly, J. F., "Eliminating the Unwanted Zero in Ones' Complement Addition," Electronics, vol. 49, no. 3, 103-105, February 5, 1976.

CONFERENCE PROCEEDINGS

McClure, R. M. and M. J. Flynn, "An Integrated Facility for Emulation Research," Proc. 2nd USA-Japan Computer Conference, Tokyo, Japan, August 26-28, 1975, 42-46.

Usas, A. M., "Fail-Safe Circuits: A Means to Improve Reliability and Maintainability of I/O Subsystems," Digest of CompCon-East, Washington, D. C., September 9-11, 1975.

Shedletsky, J. J., "A Rationale for the Random Testing of Combinational Digital Circuits," Digest of Papers, CompCon-East, Washington, D. C., September 9-11, 1975.

Rau, B. R., "A New Philosophy for Interconnection on Multilayer Boards," Proc. 13th Design Automation Conf., San Francisco, California, June 22-24, 1976, 225-231.

McCluskey, E. J., "Micros, Minis, and Networks," invited paper, Proc. 20 Years of Computer Science at U. of Pisa, Pisa, Italy, 1975.

Wakerly, J. F,, C. R. Hollander and D. Davies, "Placement of Micro-instructions in a Two-Dimensional Address Space," Proc. Eighth Annual Workshop on Microprogramming, Chicago, Illinois, September 1975, 46-51.

Wakerly, J. F., "Reliability of Microcomputer Systems Using Triple Modular Redundancy," Digest of Papers, 12th Annual Computer Society Conf. (CompCon), San Francisco, California, February 24-26, 1976, 23-26.

Widdoes, L. C., "The Minerva Multi-Microprocessor," Proc. 3rd Annual Symp. on Computer Architecture, Clearwater, Florida, January 19-21, 1976, 34-39 .

Uncapher, K. and V. G. Cerf, "The ARPANET--A User Perspective," Proc. Eighth National Data Processing Congress, Sao Paulo, Brazil, October 1975.

Dalal, Y., "Distributed File Systems," Proc. Berkeley Workshop on Distributed Data Management and Computer Networks, Berkeley, California, May 25-26, 1976.

Parker, K. P., "Compact Testing: Testing with Compressed Data," Proc. Sixth Intl. Symp. on Fault-Tolerant Computing, Pittsburgh, Pennsylvania, June 21-23, 1976.

Losq, J., "Referenceless Random Testing," Proc. Sixth Intl. Symp. on Fault-Tolerant Computing, Pittsburgh, Pennsylvania, June 21-23, 1976.

Shedletsky, J. J., "A Rollback Interval for Networks with an Imperfect Self-Checking Property," Proc. Sixth Intl. Symp. on Fault-Tolerant Computing, Pittsburgh, Pennsylvania, June 21-23, 1976.

Losq, J., "Multiple Failures and Redundant Systems," Proc. 1976 Conf. on Information Sciences and Systems, Johns Hopkins Univ., Baltimore, Maryland, March 31-April 2, 1976.

vanCleemput, W. M., "On the Topological Aspects of the Circuit Layout Problem," Proc. 13th Annual Design Automation Conf., San Francisco, California, June 22-24, 1976.

TECHNICAL REPORTS

Computer Reliability

Kolupaev, S., 'Cascade Structures in Self-Checking Networks," Tech. Rpt. no. 108, April 1976.

McCluskey, E. J., Wakerly, J. F. and Ogus, R. C., 'Center for Reliable Computing: Current Research,' Tech. Rpt. no. 100, October 1975.

Parker, K. P., 'Adaptive Random Test Generation,' Tech. Rpt. no 109, May 1976.

Shedletsky, J. J., "Error Latency of a Combinational Digital Circuit,' Tech. Rpt. no. 103, December 1975.

Shedletsky, J. J., "Error Correction by Alternate-Data Retry,' Tech. Rpt. no. 113, May 1976.

Varszegi, S., "Deterministic Sequential Networks Under Random Control,' Tech. Rpt. no. 97, September 1975.

Wakerly, J. F., "Principles of Self-Checking Processor Design and an Example," Tech. Rpt. no. 115, December 1975.

Commuter Architecture

Flynn, M. J., Hoevel, L. and Neuhauser, C., "The Stanford Emulation Laboratory," Tech. Rpt. no. 118, April 1976.

Hoevel, L. and Wallach, W. A., "A Tale of Three Emulators," Tech. Rept. no. 98, November 1975.

Polstra, J. D. and Proskurowski, A., "EMMYPL: Description and Implementation," Tech. Rpt. no. 99, November 1975.

Rao, G., "Performance Analysis of Cache Memories," Tech. Rpt. no. 110, August 1975.

Rau, B. R., "An 'Almost-Exact' Solution to the N-Processor, M-Memory Bandwidth Problem," Tech. Rpt. no. 117, June 1976.

Wallach, W. A., "High Performance Emulation," Tech. Rpt. no. 102, November 1975.

Wallach, W. A., "EMMY/360 Functional Characteristics," Tech. Rpt. no. 114, June 1976.

TECHNICAL REPORTS (continued)

Design Automation

Rau, B. R., "A New Philosophy for Wire Routing," Tech. Rpt. no. 101, November 1975.

vanCleemput, W. M., "Mathematical Models for the Circuit Layout Problem," Tech. Rpt. no. 106, February 1976.

Computer Networks

Crane, R., "Asynchronous Serial Interface for Connecting a PDP-11 to the ARPANET," Tech. Rpt. no. 116, July 1976.

Dalal, Y. K., "A Distributed Algorithm for Constructing Minimal Spanning Trees in Computer-Communication Networks," Tech. Rpt. no. 111, June 1976.

Parallel Computer Systems

Holland, W. S., "A Programmable Computer Vision System Based on Spatial Relationships," Tech. Rpt. no. 104, December 1975.

Russell, D. L., "State Restoration Among Communicating Processes," Tech. Rpt. no. 112, June 1976.

Saxena, A., "A Verified Specification of a Hierarchical Operating System," Tech. Rpt. no. 107, January 1976.

Sunshine, C. A., "Interprocess Communication Protocols for Computer Networks," Tech. Rpt. no. 105, December 1975.

TECHNICAL NOTES

Computer Reliability

Beaudry, D., "Dual Redundancy; A Survey," Tech. Note no. 93, July 1976.

Betancourt, R. and McCluskey, E. J., "Analysis of Sequential Circuits Using Clocked Flip-Flops," Tech. Note no. 82, August 1975.

Ogus, R., "Design of Fault-Tolerant Iterative Cell Arrays," Tech. Note no. 85, December 1975.

Parker, K., "Compact Testing: Testing with Compressed Data,' Tech. Note no. 64, September 1975.

Parker, K., 'Adaptive Random Test Generation," Tech. Note no. 73, October 1975.

Shedletsky, J. J., "Comment on the Sequential and Indeterminate Behavior of an End-Around-Carry Adder," Tech. Note no. 78, November 1975.

Verdillon, A., "Procedures to Obtain Optimal Test Sequences," Tech. Note no. 80, December 1975.

Wakerly, J. F., "Eliminating the Unwanted Zero in Ones' Complement Addition, Tech. Note no. 71, November 1975.

Computer Architecture

Flynn, M. J., "Studies in the Organization of Computer Systems," Tech. Note no. 69, November 1975.

Flynn, M. J., "Feasibility of Real Time Emulation," Tech. Note no. 70, November 1975.

Hedges, T. S., "EMMY/360 Cross Assembler," Tech. Note no. 74, December 1975.

Hoevel, L. and Wallach, W. A., "Proposed Enhancements to EMMY," Tech. Note no. 67, November 1975.

Neuhauser, C., "Functional Description of the EMMY Main Memory System," Tech. Note no. 57, August 1975.

Neuhauser, C., "An Emulation Oriented, Dynamic Microprogrammable Processor (Version 3)," Tech. Note no. 65, October 1975.

Neuhauser, C., "EMMY System Peripherals -- Principles of Operation," Tech. Note no. 77, December 1975.

Computer Architecture (continued)

Polstra, J., "EMMYPL User's Manual," Tech. Note no. 86, April 1976.

Wallach, W. A., "System/360 Emulator Performance Estimate," Tech. Note no. 66, November 1975.

Wallach, W. A., "EMMYXL User's Guide," Tech. Note no. 84, March 1976.

Wallach, W. A., "EMMY/UNIBUS Interface Preliminary Specification," Tech. Note no. 88, June 1976.

Wallach, W. A., "Virtual Addressing for the EMMY/360," Tech. Note no. 89, June 1976.

Yu, P., "Some Notes on the Log Conjecture for Parallel Processors," Tech. Note no. 90. June 1976.

Design Automation

vanCleemput, W. M., "Comments on 'Theory of Multiplace Graphs'," Tech. Note no. 81, December 1975.

Computer Networks

Cerf, V. G., "ARPA Internetwork Protocols Project Status Report," Tech. Note no. 68, November 1975.

Cerf, V. G., "TCP Resynchronization," Tech. Note no. 79, January 1976.

Cerf, V. G., "ARPA Internetwork Protocols Project Status Report," Tech. Note no. 83, February 1976.

Cerf, V. G., "SCCU/MCCU Characteristics for Autodin II," Tech. Note no. 92, July 1976.

Crane, R. C., "Bell 303 Modem Replacement," Tech. Note no. 91, July 1976.

Mathis, J. E., "Single-Connection TCP Specification (preliminary documentation)," Tech. Note no. 75, January 1976.

Rubin, D. E., "TELNET Under Single-Connection TCP Specification," Tech. Note no. 76, February 1976.

TECHNICAL NOTES (continued)

Parallel Computer Systems

Hollander, C. R., "Correctness Condition Graphs," Tech. Note no. 52, September 1975.

Hollander, C. R., "Construction of Primitives for a Multiprocess Environment," Tech. Note no. 63, August 1975.

Saxena, A., "An Efficient Implementation of Monitors and Condition Variables," Tech. Note no. 72, August 1975.

PAPERS ACCEPTED FOR PUBLICATION

vanCleemput, W. M., "Mathematical Models for the Circuit Layout Problem," IEEE Trans. on Circuits and Systems, to appear September 1976.

vanCleemput, W. M., "Hypergraph Models for the Circuit Layout Problem," to appear in Applied Mathematical Modelling.

Wakerly, J. F., "Checked Binary Addition Using Check Symbol Prediction and Checksum Codes," to appear in J. Design Automation and Fault-Tolerant Computing, vol. 1, no. 1, October 1976.

Parker, K. P., "Adaptive Random Test Generation," to appear in J. Design Automation and Fault-Tolerant Computing, vol. 1, no. 1, October 1976.

Flynn, M. J. and R. Kosaraju, "Processes and Their Interactions," to appear in Kybepnhthe (Special Issue), vol. 5, 1976.

PAPERS PRESENTED AT MEETINGS

Cerf, V. G., "Sequencing and Congestion Control in Packet Networks," invited paper, Joint IFIP/IIASA Workshop on Data Communications, Intl. Inst. for Applied Systems Analysis, Luxenburg, Austria, September 15-19, 1975.

Cerf, V. G., "State-of-the-Art of Communication Protocol Design," invited speaker, NSF Symp. on Computer Networking, George Washington U., Washington, D. C., March 18-19, 1976.

PAPERS PRESENTED AT MEETINGS (continued)

Rau, B. R., "Analytic Performance Evaluation of Memory Hierarchies," IEEE Lake Arrowhead Workshop on Advances in Storage for Minis and Micros, Lake Arrowhead, California, September 3-5,1975.

PAPERS ACCEPTED FOR PRESENTATION

Saxena, A. R. and T. H. Bredt, "Verification of a Monitor Specification," accepted for presentation at 2nd Intl. Conf. on Software Engineering, San Francisco, California, October 13-15, 1976.

Phillips, J. V. and T. H. Bredt, "Design and Verification of Real-Time Systems," accepted for presentation at 2nd Intl. Conf. on Software Engineering, San Francisco, California, October 13-15, 1976.

Dalal, Y., "A Distributed Algorithm for Constructing Minimal Spanning Trees in Computer Communication Networks," invited paper, to be presented at Fifth Texas Conf. on Computing Systems, Austin, Texas, October 18-19, 1976.

Karp, R. A. and D. C. Luckham, "Verification of Fairness in an Implementation of Monitors," accepted for presentation at 2nd. Intl. Conf. on Software Engineering, San Francisco, California, October 13-15,1976.

Widdoes, L. C., "Architectural Considerations for General Purpose Multiprocessors," accepted for presentation at IEEE Computer Society CompCon-East, Washington, D. C., September 8-10, 1976.