NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Technical Report 32-1409

Design Methods for Fault-Tolerant Navigation Computers

Algirdas Avižienis

JET PROPULSION LABORATORY
CALIFORNIA INSTITUTE OF TECHNOLOGY
PASADENA, CALIFORNIA

October 15, 1969

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Technical Report 32-1409

Design Methods for Fault-Tolerant Navigation Computers

Algirdas Avižienis

JET PROPULSION LABORATORY
CALIFORNIA INSTITUTE OF TECHNOLOGY
PASADENA, CALIFORNIA

October 15, 1969

Prepared Under Contract No. NAS 7-100
National Aeronautics and Space Administration

Preface

The work described in this report was performed by the Guidance and Control Division of the Jet Propulsion Laboratory.

Acknowledgment

The design of the STAR computer was accomplished by the Flight Computers and Sequencers Section of the Guidance and Control Division. Thanks are due to many of its members for their support and advice. Major contributions to the development of the STAR Computer have been made by F. P. Mathur, D. A. Rennels, J. A. Rohr, P. H. Sobel, J. J. Wedel, and A. D. Weeks. The power switch has been developed by the Stanford Research Institute, Menlo Park, California, and a fault-tolerant Read-Only Memory has been designed by the M.I.T. Instrumentation Laboratory, Cambridge, Massachusetts, under subcontracts from JPL. Construction of the STAR computer breadboard has been carried out by J. Buchok, N. Funsten, J. Schooler, and B. Stall.

Contents

Figures

Abstract

This report discusses the problem of fault tolerance in digital computers with strong emphasis put on computers used in on-board navigation systems of aircraft and spacecraft. An extension of automatic maintenance to an entire navigation system is introduced.

Two principal approaches to fault tolerance in navigation computers are analyzed. Examples are taken from recent aerospace computers, i.e., the *Saturn V* computer, and the Self Testing And Repairing spacecraft guidance computer. The problem of fault tolerance is identified and defined, and current applications of the computers are discussed. Some current exploratory research and research objectives for the future are described.

Design Methods for Fault-Tolerant Navigation Computers

I. Introduction

Computational errors in a computer are caused by logic faults, which are deviations of logic variables from their prescribed values. Transient and permanent faults are caused by component failures, intermittent malfunctions, and external interference during computation. Error-free computation in the presence of various faults is accomplished by a fault-tolerant computer using protective redundancy. Protective redundancy may be introduced in the form of additional programs (software redundancy), repetition (time redundancy), and additional components (hardware redundancy). Hardware redundancy is applied to provide fault tolerance when navigation computers are required to carry out real-time computation without the possibility of maintenance operations by a human operator. Two principal approaches are masking redundancy (structural replication), and standby redundancy (automatic maintenance).

II. The Fault Problem in Digital Computers

The idealized or "perfect" digital computer can serve as a reference point in the discussion of fault tolerance. Two-state logic circuits (storage elements and operator elements) are the elementary building blocks of a digital computer. The logic design of the computer specifies how these circuits are interconnected and which one of the two possible logic values ("true" and "false", most frequently designated by "1" and "0" respectively), will exist at the input and output points of every logic circuit at every defined instant of time. The "perfect" computer always functions according to the specifications of its logic design.

It has been observed, however, that digital computers, constructed for use in various applications, occasionally deviate from the design specifications. The logic value(s) at one or more points of the computer logic become opposite to the specified value(s). The deviation of a logic variable from its prescribed value is called a logic fault, or more concisely, a fault. Most faults will cause an error in the program being executed by the computer: either an instruction is not executed correctly, or an incorrect result is computed. Both types of errors may be caused at once by some faults. The exact nature of a fault depends on whether a permanent failure, or a temporary (transient) malfunction of one or more components (resistors, transistors, connections, etc.) of the computer is the cause.

Transient faults are temporary deviations of logic values from design specifications. Two main causes of transient faults are intermittent component malfunctions and external interference with the operation of the computer. Such interference is caused by irregularities of the

power supply, stray electromagnetic radiation, severe environments, and similar events. Transient faults cause errors in computation without leaving a permanent record, and their occurrence will not be detected by the periodic checkouts of the computer, that are sufficient to detect the presence of permanent faults.

Permanent faults are caused by permanent component failures. As a result the logic value at a certain point remains constant ("stuck on zero", or "stuck on one") regardless of design specification. Beside such determinate faults, occasionally an indeterminate (or "stuck on X") permanent fault may occur, in which the logic value varies between "1" and "0" but not according to the design. Such faults can be caused by drift of component values and similar phenomena.

Logic faults also differ as to the extent to which they affect the computer. Independent, or local, faults affect only one logic circuit in the computer. They are caused usually by random failures of the components. Related, or catastrophic, faults simultaneously affect two or more logic circuits. They are very likely to occur in the case of physical damage or external interference to the computer. The recent advances in the large-scale integration of electronic circuits lead to very close placement of logic circuits and make related faults more probable than is the case when discrete components are used.

III. The Application of Protective Redundancy

The preceding discussion of logic faults now permits the definition of a fault-tolerant computer. It is a computer that can carry out error-free programs in the presence of logic faults (Ref. 1). The classes of faults that are tolerated and the parts of the computer in which they may occur must be identified in every claim of fault tolerance. Fault tolerance in digital computers is attained by means of protective redundancy. In this report, protective redundancy is defined as all additional programs (soft-ware redundancy), repetition of operations (time redundancy), and additional components or circuits (hardware redundancy), that are not needed in the "perfect" computer, but serve to provide fault tolerance in the physical implementation of the same logic design.

Software redundancy includes emergency action programs that are used when a fault has been detected. Also included are diagnostic programs that are executed (periodically or upon request) to test all logic circuits of the computer for the presence of permanent faults.

Time redundancy includes the repetition of a program or a segment of a program after a fault has been detected (or suspected). The repetition of programs to compare the results, the inclusion of duplicate instructions, and the "reasonableness checks" also fit into this category.

Hardware redundancy includes the components and/or circuits that serve to provide fault tolerance. Two distinct approaches to the use of hardware redundancy are masking and recovery. The masking, or static, approach uses a "massive" replication of each component or circuit to two or more copies. All copies are permanently connected and receive power. Component failures or logic faults are masked by the presence of other copies of the same item. The fault masking occurs instantaneously and automatically; however, if the fault is not susceptible to masking and causes an error, a delayed recovery is not possible in the masking scheme. The two most used variants of masking are component quadding for individual electronic components (Fig. 1) and triple modular redundancy (TMR) with voting (Fig. 2) for logic circuits or larger parts of a computer. Several other variants of masking redundancy have been studied but were not found prac-

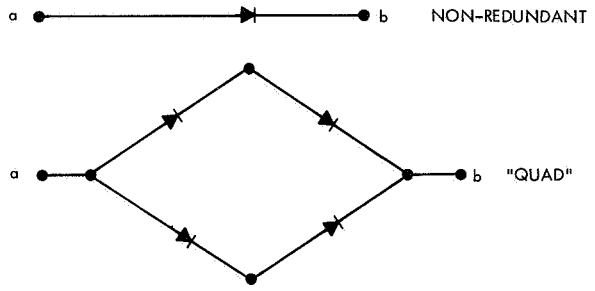

Fig. 1. "Quadded" components (diodes)

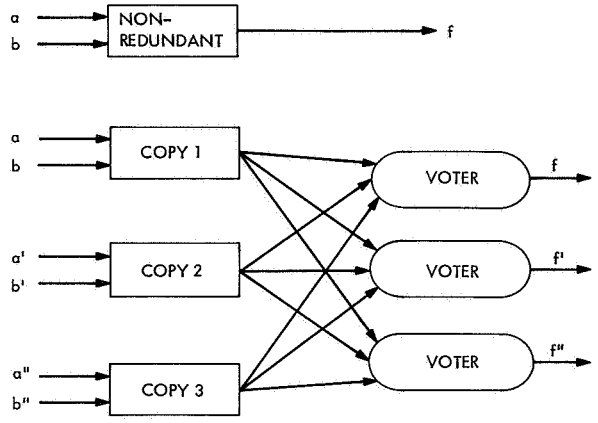

Fig. 2. Triple-modular redundancy with voting

tical because of various drawbacks (excessive cost and lack of practically realizable special components).

The recovery, or dynamic, approach in hardware redundancy requires two consecutive actions. The presence of a fault is first detected, then a recovery action either eliminates the fault or corrects the error which was caused. The redundancy is usually introduced in a selective, rather than massive, fashion. The means of error detection are error-detecting codes, monitoring circuits, synchronization checks, duplication, and comparison of critical functions. Software and time redundancy are also applicable to the error detection. The recovery may be a repetition of a program segment, the application of an error-correcting code, or the replacement of a faulty part by a spare.

The reconfiguration of a computer into a new system without the failed part has also been investigated. It must, however, be recognized here that the computing capacity of the system has been reduced, therefore the fault has been only partially tolerated. Such computers can be designated as "partially fault-tolerant" and, therefore, not suitable in applications that require a constant computing capacity for a prescribed time.

IV. Current Methods of Fault Tolerance in Aerospace Computers

The discussion is now directed to the special problems of introducing fault tolerance into aerospace computers that carry out computations required for the navigation and control of aircraft, spacecraft, and related vehicles. On many occasions, the same computer is programmed to perform the reduction and storage of telemetry and scientific data, as well as to monitor the performance of various other subsystems. The choice among the methods of fault tolerance is strongly affected by these requirements and by the constraints imposed by the overall system design.

Ground-based digital computer installations depend on extensive diagnostic programs and on the availability of maintenance experts to restore proper functioning after a fault has been detected. In cases where a longer interruption cannot be tolerated, a second computer is held in readiness with the same set of programs as the first. Intermediate results of computation are periodically transferred from the primary to the backup computer memory. As long as the primary computer functions correctly, the backup unit performs auxiliary computing. When an error in the primary unit is detected, the backup unit assumes the role of the primary unit. In real-time operation, both computers carry out the primary programs, and frequent comparisons are used for error detection. The Electronic Switching System (ESS) No. 1 of Bell Telephone is a good example of a real-time, special-purpose computer with duplication and extensive diagnostic programs.

The typical aerospace computer operates in a more constrained environment than the ground based installation. A human maintenance specialist is not available during an actual mission (the interval of time $0 \leq t \leq T_m$ during which the computer must remain operational) nor is storage for large diagnostic programs available. Computing at the full capacity and speed is required at critical times during the mission, with one of these times usually coming near the mission's end. There are strict power, weight, and volume constraints to be observed in this environment. An example where all of these limits are encountered is in the computer aboard an unmanned space vehicle intended for the exploration of the outer planets of the solar system.

The first extensive application of protective redundancy (of the masking type) in an aerospace computer is found in the Primary Processor and Data Storage Unit of the Orbiting Astronomical Observatory (OAO), (Ref. 2). Discrete electronic components were used in this project, which was initiated in 1960. The individual components of each logic circuit are replaced by two or four copies each, arranged in parallel, series, or "quad" (Fig. 1) arrays. This approach uses structural redundancy at the lowest level at which replication is practical. The choice of this method was based on the initial reliability goal of 0.95 for one year (8760 h) operation in space and a 15-minute launch interval during which failure rates are assumed to be 1000 times higher than during actual operations in space, and on the known failure rates of the components. A pre-launch period of 3240 h was added subsequently to the specification. The resulting reliability prediction was about 0.75 for the entire period of operation (including data storage). The complete Primary Processor and Data Storage unit contains approximately 70,000 component parts and approximately 212,000 magnetic cores for data storage. The Data Storage Unit uses storage in quadruplicate for command words (256 locations) and in duplicate for data words (8192 locations). Storage redundancy is needed because some memory circuits require high precision and cannot be protected by component replication, which requires wider circuit tolerances.

A second important use of protective redundancy is found in the guidance and control computer for the *Saturn V* launch vehicle (Ref. 3). The design of this computer was initiated in 1961, with deliveries required for the 1964–1968 period. The specified reliability goal was 0.99 for a mission time of 250 h and it led to the choice of triplication of the logic elements with voting at selected locations (Fig. 2). An exception was the core memory that was protected by duplication, parity checking for error detection, and monitoring of drive currents by special circuits. The TMR part of the computer consists of seven modules averaging 13 voted outputs for each module. Extensive computation of the reliability using a Monte Carlo simulation was used to verify the adequacy of the design, because the redundant system is too complex for an analytic calculation of the reliability. The results indicated a 250-hour reliability of 0.9992 for the TMR portion of the computer which had a predicted reliability of 0.973 for the non-redundant design. Ground maintenance and checkout are facilitated by the attachment of disagreement detector (DD) circuits to the voting circuits. A vote of 2 to 1 is indicated by the DD and identifies a triplet of modules in which one module has failed and is being masked by the other two.

V. Hardware Redundancy Methods: A Comparison

It is noted that the first two cases of redundant design used the masking rather than the recovery approach in the processors, as well as a complete word replication in memory. Both processors do not need error-detecting circuitry and have immediate fault-masking without interruption of operation. The conversion from a non-redundant to the "massively" redundant processor is therefore straightforward. Conversion is accomplished when either the individual circuits are redesigned with replicated components or locations for voters are chosen and the voters are designed. The application of the recovery approach is more complicated than this conversion. New functions of detection and recovery must be incorporated into the computer. The following discussion relates the problems encountered in the use of masking redundancy in aerospace computers and compares it to the recovery method.

The two computers described in the preceding section represent current application of hardware redundancy. The usefulness of the component redundancy approach of the OAO depends on the validity of the assumption of independent failures in the components forming a "quad." Miniaturization and integration of electronic circuitry places the components very close together and therefore largely invalidates the independent failure assumption. Another drawback is the very difficult circuit design problem. Proper operation must continue after a failure causes a change in circuit parameters that may shift the operating point or increase dissipation in semiconductors of the circuit. The redundant circuits dissipate more power, are slower, and need higher-precision components when the same performance is required as is of a non-redundant circuit. Pre-mission maintenance and checkout of component-redundant systems also present problems. The life of the system begins with the interconnection of the redundant circuits to form a system. The low-level masking makes component failures undetectable at the system outputs until a complete circuit failure occurs and leads to a system failure. Mission times must be defined as including the entire time interval between assembly and the end of the mission. These serious drawbacks make the extensive use of component redundancy unlikely in the future. An exception to this statement may occur in relation to a few critical circuits of a system that cannot be protected by other methods. These circuits can be built of discrete components and replaced with a new assembly before the start of each mission. Another application of component redundancy is possible when non-electronic, for example fluidic, logic circuits are used.

The TMR method of protective redundancy avoids several of the drawbacks of component redundancy. Circuits remain internally non-redundant, and computing occurs in three identical channels which can be adequately separated in order to retain the independent failure assumption. Pre-mission maintenance and checkout is facilitated by the use of disagreement detectors at the voters. The validity of the independent failure assumption critically depends on the inter-channel isolation and synchronization which must be provided at each voter which receives inputs from all three channels. A fault which can affect two adjacent channels through a voter, or by any other path, is not masked and will cause a system failure. The initial reliability in a TMR system is high, but the number of independent failures which it can tolerate before failing is much smaller than a component-redundant computer, and the reliability drops sharply after the initial period. The difference is emphasized by the comparison of the 8760-h mission of the OAO to the 250-h mission of the *Saturn V*.

An alternate approach which competes with masking redundancy is a replacement system (RS). The RS provides the required computing capacity as a "standard"

computer which consists of a number of operating functional units (arithmetic, memory, etc.). Each unit has one or more spares which are held in a standby (usually unpowered) condition. The RS also has a "monitor unit" which detects the presence of a fault in any operating unit and carries out the recovery operation. Transient faults must be identified and their effects corrected by repetition of a computation. Permanent faults are corrected by the replacement of the faulty unit. It is disconnected and the unit spare is switched into the standard computer. Several advantages of the RS over the masking approach are shown in the use of aerospace computers:

(1) All spares of a unit are utilized before system failure occurs.

(2) System reliability will increase if unpowered spare units have a lower failure rate (usually, some failure modes are not present when power is not applied), and parts of the standard computer are shut down during idle periods.

(3) Power requirements are lower because only one copy of each replaceable unit is powered.

(4) The system contains a built-in capacity for premission maintenance and checkout.

(5) The number of spares for a given unit may be varied as a function of the unit's reliability estimate, the mission time, and the weight and volume constraints, while the power requirement is not changed.

(6) Replacement by switching and power removal can provide strong fault isolation between units.

All of the above listed advantages are based on the existence of the monitor unit (often called the "hard core") which carries out fault detection and recovery operations with a sufficiently high reliability and without causing intolerably long interrupts in the current computation. The lack of such monitors has been the main reason why the advantages of the RS have not been realized in current aerospace computers.

VI. The STAR Computer: An Experiment in Self-Repair

The advantages of a replacement system are vital to the unmanned spacecraft with the mission of exploring the remote planets of the solar system. These advantages and the limitations of masking redundancy motivated a long-range research program concerned with the realization of a complete replacement system. Research on fault-tolerant computers was initiated at the Guidance and Control Division of Jet Propulsion Laboratory in 1961 and has culminated in the design and construction of an experimental breadboard replacement system called the "STAR" (*S*elf-*T*esting-*A*nd-*R*epairing) computer. The first short program was successfully executed by a minimal subset (the "STARlet") of the STAR computer on March 24, 1969. Full operation is planned for September 1969.

The block diagram of the STAR computer is shown in Fig. 3. The replaceable units of the system are shown in a circular arrangement. Communication between the units is carried out on two four-wire busses: the Memory Out-Bus, and the Memory In-Bus. The three-letter abbreviations designate the following units:

(1) IOP is the Input/Output Processor.

(2) IRP is the Interrupt Processor.

(3) TIP is the Time Processor.

(4) COP is the Control Processor.

(5) LOP is the Logic Processor.

(6) MAP is the Main Arithmetic Processor.

(7) ROM is the Read-Only Memory (16384 words).

(8) RWM is one Read-Write Memory module (4096 words).

At least two RWM modules are powered in order to provide redundant storage. As many as 12 RWM modules are directly addressable and may be designated under program control to operate in non-redundant, duplicated, or triplicated modes.

The monitor of the STAR system is shown in the center of the circle in Fig. 3 and is designated as TARP (*T*est-*A*nd-*R*epair-*P*rocessor). The TARP monitors the operation of the STAR computer by two methods. In the first method, an error-detecting code has been applied to all data and instruction words which are transmitted on the Memory In-Bus and Memory Out-Bus. The TARP contains two "bus checkers", which test every word for validity of its code. In the second method, a four-wire unit status code is received on the Status Lines from each powered unit. The six unit status codes are listed next in order of increasing priority. ON indicates that the unit is powered. ACTIVE occurs when at least one "Output" line has an active (logic "one") output. COMPLETE is

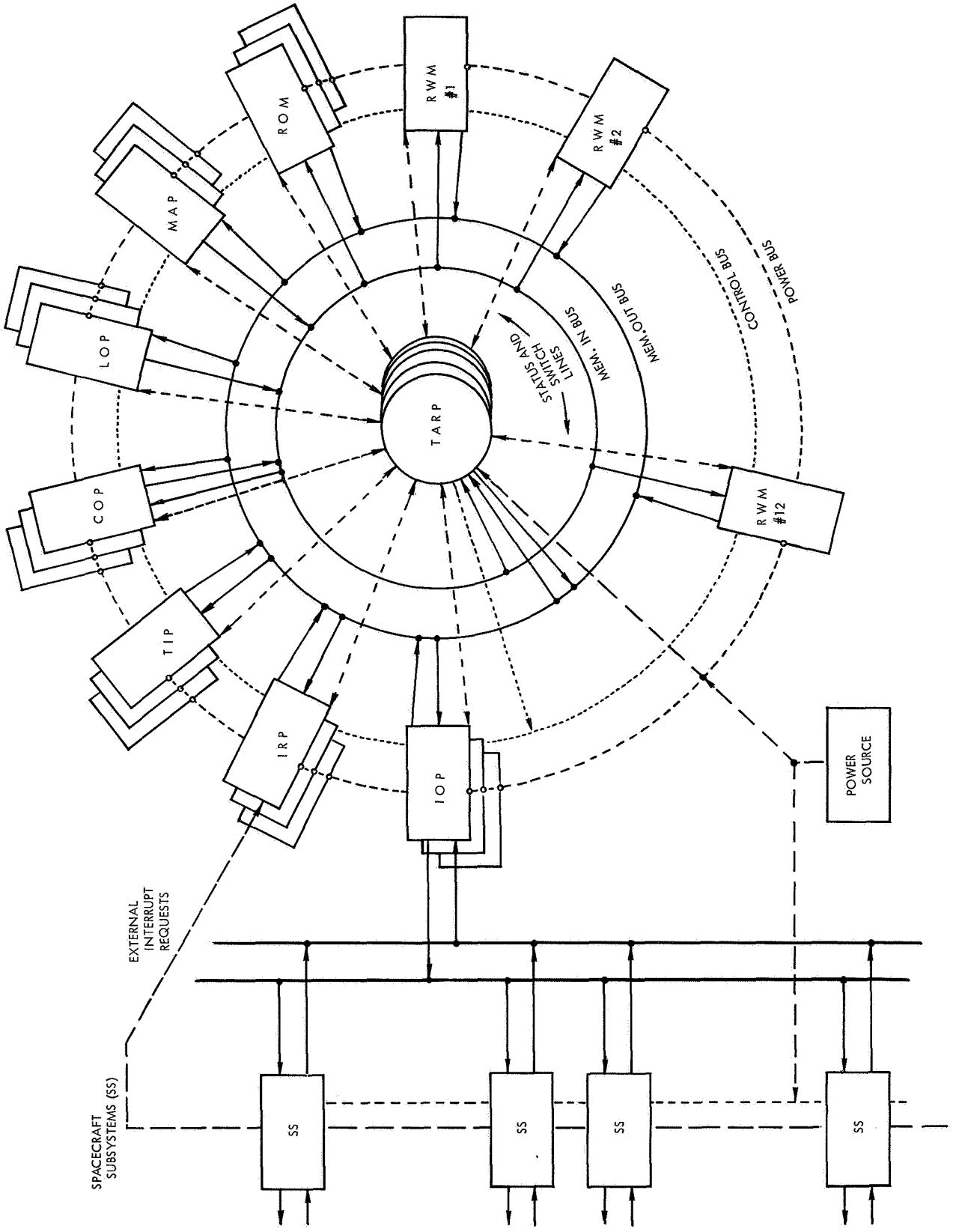

Fig. 3. The STAR computer

issued at the end of every algorithm. SPECIAL designates an event specific to the given unit. WARNING indicates the presence of an error in the program. FAULT is issued when the internal monitoring circuits of the unit detect an abnormal hardware condition. By recording the current instruction and its location, the TARP internally reconstructs the proper status for each unit. A deviation from normal operation causes an interruption of normal computing and an entry into the recovery mode.

During normal operation, the TARP supplies two signals to the Control Bus which is connected to all units: the CLOCK (1 MHz) signal and the SYNC signal. The SYNC signal indicates the beginning of every new 10-step cycle. When a fault condition occurs, its source and type is recorded in the TARP, and a RESET signal is issued on the control bus to all units. The RESET causes all powered units to reset to a standard (starting) state. The TARP then issues an unconditional transfer instruction and a segment of the current program is repeated. If the fault recurs, the faulty unit has its power switched off, and power is turned on in its next spare. Another RESET is then issued. In order to perform the repetition of a segment of the current program, the STAR computer has an instruction which stores a "rollback" address in the TARP. At the time of setting the "rollback" register, all computer words that will be needed by the program in the future must be in nonvolatile and redundant storage (separate memory modules). The TARP also has registers for the storage of the present instruction, its location, the RWM module assignments, and the source and type of an observed fault. For cases of temporary power loss and irresolvable fault conditions, the TARP contains a wired-in "Disaster Restart" procedure which begins with all other units in unpowered condition.

Both the instructions and the operands of the STAR computer are encoded in an error-detecting code. The 32-bit instruction word consists of a 12-bit operation code and a 20-bit address part. The address part is encoded in the residue code with the check modulus 15. An address part consists of the 16-bit binary address a and the 4-bit check symbol $c(a)$. The check symbol $c(a)$ has the value

$$c(a) = 15 - 15 \mid a$$

where $15 \mid a$ means "the modulo 15 residue of a." The operation code is divided into three bytes of four bits each, which are protected by the 2-out-of-4 encoding. The numeric operands are also 32 bits long and consist of a 28-bit binary number x and a 4-bit check symbol $c(x)$ which is obtained the same way as $c(a)$. The initial STAR computer design used product-coded operands $y = 15x$; however, multiple-precision considerations caused a change to the modulo 15 residue code. An extensive study of arithmetic codes (Refs. 4 and 5) preceded the design and demonstrated that these codes need only a very simple Bus Checker for validation and have fully satisfactory error detection properties.

The TARP is the "hard core" of the system. Three fully powered copies of the TARP are operated at all times together with n standby spares which are partially powered and store critical TARP data (rollback points, etc) in nonvolatile storage. All outputs of the TARP are decided by a two-out-of-$(n + 3)$ threshold vote. When one powered TARP disagrees with a voted output, it is immediately returned to the standby condition and one of the standby units receives full power and joins the powered triplet. The removed TARP unit becomes a standby unit, and its later reuse succeeds if previous removal had been caused by a transient fault. Two spares ($n = 2$) are used presently. Design effort has been concentrated on reducing the TARP to the least possible complexity. The replacement of fault units is commanded by the TARP vote and is implemented by power switching. Magnetic power switches have been developed which are part of each unit's power supply and are designed to open for most internal failures. The information lines of each unit are permanently connected to the busses through isolation circuits. The signal on the bus is the logic "OR" function of all inputs from the units. The power switch and the bus utilize component redundancy and electronic design for protection against fatal "shorting" failures.

The STAR computer is a replacement system that employs a balanced mixture of coding, monitoring, standby redundancy, replication with voting, and repetition to attain hardware-controlled self repair and protection against transient faults. The principal objective of the design is to attain fault tolerance for the widest possible variety of faults (i.e., transient, permanent, random, and catastrophic). The computer is intended to serve as a vehicle for further experimentation and refinement of the recovery techniques, (especially in the protection of the TARP and in the design of buses and power switches). An extensive testing and validation program is being planned for the STAR breadboard. A better understanding of recovery from transient faults will be sought by actual injection of such faults and observation of the recovery procedure.

The investigations of the STAR computer have also stimulated new research efforts in fault tolerance. Current investigations are concerned with the following areas:

(1) Hardware-software interaction in a fault-tolerant system with recovery, such as the interaction of the TARP and the operating system, the automatic verification and insertion of "rollback" points in all programs, and auxiliary diagnostic programs.

(2) Studies of advanced recovery techniques, i.e., post-catastrophic restart, TARP replacement schemes, recovery from massive interference.

(3) Computer-aided reliability prediction for STAR-like fault-tolerant systems.

(4) Advanced component technology, especially methods to attain bus and power switch immunity to faults.

(5) Formulation of a theory of fault-tolerance by interpretation of extensive experiments with the STAR breadboard as the instrument.

VII. Automatic System Maintenance: An Extension of the STAR Computer

The existence of a self-repairing computer aboard a spacecraft suggests the systematic extension of the same techniques to other parts of the entire spacecraft system. The role of the monitor now is assigned to the entire STAR computer that stores the diagnostic and emergency action programs for other subsystems, (actuators, sensors, transmitters, and scientific instruments). Unpowered spares are stocked for these subsystems, and power switching is utilized to implement a replacement when the diagnostic programs indicate that the performance is below a specified standard. The advantages of a replacement system over masking redundancy apply here as well, especially the power requirement for single copies only, the utilization of lower failure rates of unpowered spares, and the ability to use all the copies of each unit.

We use the monitoring of the propulsion system performance in a spacecraft as an example of the application of automatic maintenance principles. The sensors which provide data to the computer indicate the pressures in the fuel line, the oxidizer line and in the combustion chamber, as well as the combustion chamber temperature. If an abnormal fuel or oxidizer flow condition is detected, the motor is shut down and an alternate valve is operated. A record of the event is stored for future reference and for telemetry. The chamber temperature and the accelerometer outputs are also monitored by the computer, and if abnormal conditions are detected, an adjustment is made or the motor is shut down, as dictated by the maintenance program. Another example of automatic maintenance is the monitoring of gas leakage in the attitude-control system. When excessive gas leakage is detected a standby valve can be employed to replace the leaking valve.

The application of automatic maintenance offers a significant improvement in the expectation of mission success and should also be useful in many other critical applications. High-speed aircraft and automated systems for the monitoring of patients in hospitals of the future are examples of this usage.

References

1. Avižienis, A., "Design of Fault-Tolerant Computers," *AFIPS Conference Proceedings*, Vol. 31, pp. 733–743, Fall Joint Computer Conference, Anaheim, Calif., Nov. 1967.

2. Lewis, T. B., "Primary Processor And Data Storage Equipment For The Orbiting Astronomical Observatory," *IEEE Trans. EC-12*, pp. 677–686, 1963.

3. Anderson, J. E., and Macri, F. J., "Multiple Redundancy Applications in a Computer," *Proc. 1967 Annual Symposium on Reliability*, pp. 553–562, Washington, D.C., 1967.

4. Avižienis, A., "Concurrent Diagnosis of Arithmetic Processors," *Conference Digest of The 1st Annual IEEE Computer Conference*, pp. 34–37, Chicago, Ill., 1967.

5. Avižienis, A., "The Diagnosable Arithmetic Processor," in *Supporting Research and Development*, Space Programs Summary 37-37, Vol. IV, pp. 76–80. Jet Propulsion Laboratory, Pasadena, California, 1966.

6. Aldrich, W. H., and Alonso, R. L., "The 'Braid' Transformer Memory," *IEEE Trans. EC-15*, pp. 502–508, 1966.