

M. Y. Hsiao
W. C. Carter
J. W. Thomas
W. R. Stringfellow

Reliability, Availability, and Serviceability of IBM Computer Systems: A Quarter Century of Progress

Computer systems have achieved significant progress in the areas of technology, performance, capability, and RAS (reliability/availability/serviceability) during the last quarter century. In this paper, we shall review the advances of IBM computer systems in the RAS area. This progress has for the most part been evolutionary; however, in some cases it has been revolutionary. RAS developments have been driven primarily by technological advances and by increases in functional capability and complexity, but RAS considerations have also played a leading role and have improved technological and functional capability. The paper briefly reviews the progress of computer technology. It points out how IBM has maintained or improved its systems RAS capabilities in the face of the greatly increased number of components and system complexity by improved system recovery and serviceability capability, as well as by basic improvements in intrinsic component failure rate. The paper also covers the CPU, tape, and disk areas and shows how RAS improvements in these areas have been significant. The main objective is to provide a comprehensive view of significant developments in the RAS characteristics of IBM computer systems over the past twenty-five years.

Introduction and general concepts

Reliability is a measure of the consistency with which a system successfully provides its specified services. Serviceability is a measure of the ease with which the system is restored to its specified state. Availability is the percentage of the time during which the system is providing that specified service [1]. The characteristics of and the effect on the system with regard to these three interrelated quantities are referred to as the *system RAS*.

The central issue in designing systems with good RAS characteristics is *recovery*—reduction of fault occurrence, detection and counteraction of errors [2], and efficient repair procedures. Recovery implies resumption of operation with data integrity. Figure 1 illustrates the basic relationship between faults and system RAS for a unified hardware/system of RAS. In the center circle,

system faults may be caused by the intrinsic device failure rate, by design faults, or by outside interference. When the fault causes an error, the first line of defense is error detection, followed by error correction (usually with error-correction codes) or by retry. If the erroneous effect of the fault no longer exists, operation continues without repair in the reliable state. If these mechanisms do not work, the effects of the error propagate to a subsystem, and error recovery usually proceeds using an error-recovery program, with deletion of the offending subsystem. If the error cannot be contained within a subsystem, other methods of correction are needed, possibly with human intervention. However, the system still provides some service. Finally, the system may not be able to proceed at all and immediate repair is necessary. In this case, serviceability is important for efficient restoration of service.

Copyright 1981 by International Business Machines Corporation. Copying is permitted without payment of royalty provided that (1) each reproduction is done without alteration and (2) the *Journal* reference and IBM copyright notice are included on the first page. The title and abstract may be used without further permission in computer-based and other information-service systems. Permission to republish other excerpts should be obtained from the Editor.

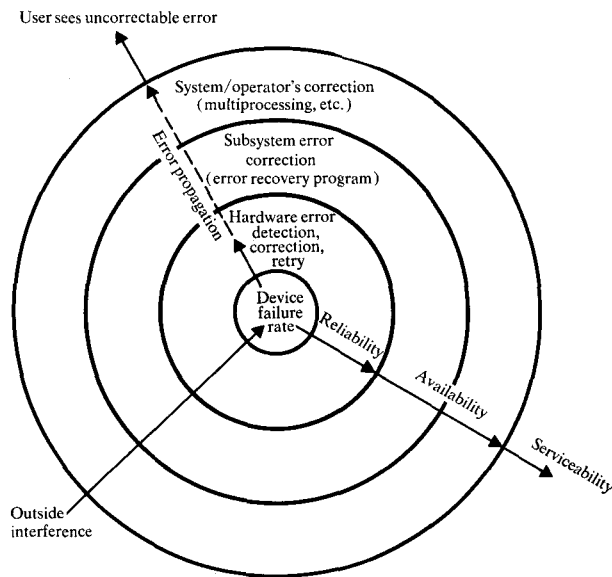


Figure 1 System reliability, availability, and serviceability.

In the following sections of this paper, we will first briefly examine the trends of hardware technology and their effect on RAS. Then we shall treat system RAS and show how the basic ideas just described have been implemented. We will then examine the CPU and discuss the special RAS features that have been added. We will also consider magnetic storage and show how the reliability of tape and disk devices has been markedly improved. Although this paper includes the more significant advancements and trends associated with RAS in IBM computer systems, it is not intended to be a complete history or to include the comprehensive progress of all systems, boxes, and devices. For example, the RAS advances of the IBM Federal Systems Division's products/systems are not covered here. The papers by James [3] and by Olsen and Orrange [4] touch on this area, while that by Jarema and Sussenguth [5] includes various error detection/correction codes as well as checking and diagnostic features of data communications systems.

Technology, system, and service trends

During the last twenty-five years, the computer industry has made tremendous progress in its technology and functional capability. Technological progress not only offers cost and performance improvement, but also provides significant advances in RAS. In the past decade, progress in technology from discrete components to LSI has been so rapid that it is frequently referred to as a revolution.

The earliest commercially available products for data processing used vacuum tubes to perform logic, and these products packaged one or more vacuum tubes along with associated passive components in a *field-replaceable unit* (FRU). Logic functions were performed entirely by the vacuum tubes. As product architectures increased in complexity, germanium diodes replaced the vacuum tubes for much of the logic switching. The FRUs increased in size and housed several vacuum tubes along with associated diode switches and passive components.

Maintenance for these systems followed the tradition of the time in isolating faults. The *customer engineer* (CE) was expected to have a comprehensive knowledge of the logic of the system as well as an understanding of the logic circuits. He was provided a complete set of logic diagrams and detailed electrical diagrams for each of the FRUs. Since there was *minimal hardware checking* in these early products, the errors usually had to be recreated with exercise programs so that fault isolation could take place. Two methods of fault isolation were used; the first was usually substitution of a spare FRU for the suspected unit. If this was unsuccessful, the use of logic diagrams and an oscilloscope for signal tracing was invoked to the failing condition. The most difficult problems to resolve were those that could not be recreated and would only occur during the customer's operation. Analysis depended on the ingenuity of the CE along with a very detailed knowledge of the product and the customer's application.

The vacuum-tube FRU was usually repaired in the field by either replacing the vacuum tube or one of the passive components. The use of transistors initiated the pluggable printed-circuit card as the new field-replaceable unit. For reliability reasons, no sockets were provided for any of the components on these cards, including the transistors; therefore, the CE was no longer expected to repair the FRUs. This was the first step in a trend that was continued with each new technology generation. The separation between the logical elements and the CE has widened with each significant increase in logic density. The early transistor technologies were made up entirely of discrete components housed on printed cards that were plugged into a back panel and interconnected by means of wire-wrapped connections. As a result of the higher density and lower cost of semiconductors, error checking was becoming more prevalent, and we began to see use of error-correcting codes in our most powerful system, the 7030 (Stretch).

Accompanying these improvements was a significant reduction in the failure rates of the logic and storage elements. These trends are shown in Figs. 2(a) and (b). The

reduction in the number of interconnections required contributed significantly to this reliability improvement; the environment in which the semiconductors reside is more protective as levels of integration increase.

Serviceability is enhanced by the inclusion of low-cost error-detection hardware, which becomes economically feasible with higher levels of integration. With error detection and logout, error re-creation is avoided; this is an important advantage in the fault-isolation process. Fault isolation is further enhanced by increases in the number of circuits contained in the FRUs. The number of logic elements on the FRU has gone from one circuit to ten thousand on current products. Replacement of one FRU has a higher probability of a successful repair as the number of FRUs making up a product decreases.

Higher levels of integration and larger numbers of logic elements on FRUs have decreased the number of test points available. As the availability of test points has decreased, using an oscilloscope and logic diagrams to understand and verify the operation of a FRU has become impractical. This has minimized the need for discretionary judgment on the part of service personnel and has shifted greater responsibility to the product developer to provide a highly serviceable product.

Technological advances have made practical the use of service processors either integrated into the product or as portable units brought to the product by service personnel. These are highly flexible minicomputers that are connectable via a service interface on the using product. Use of these service processors can significantly enhance the effectiveness of on-site personnel. It is also possible to provide similar visibility by way of a teleprocessing link into the system from a remote support capability, allowing a more experienced person or an engineer to assist in the solution of a problem without the delay of travel.

In the early 1960s, the IBM Field Engineering Division, while taking into account the growing complexity of hardware and software, concluded that a technical data access system could be of significant value. It was proposed that much of the rediscovery time by the CE could be saved if data files were compiled on difficult hardware and software problems and their associated symptoms. This would be accomplished by way of a terminal and teleprocessing network with access to a central data bank. Access to the network was to be made available to technical information center personnel contacted by CEs with unresolved problems. Using the product type as an index, it was possible for the information center people to access all of the symptom faults in the data bank for the product

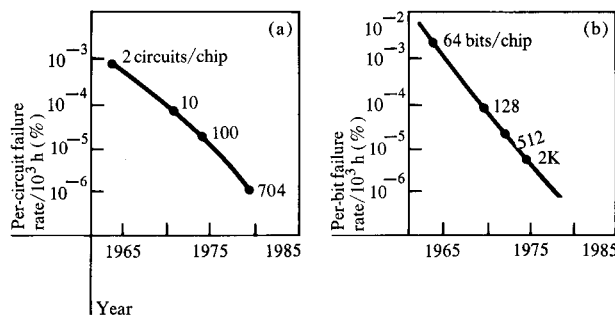


Figure 2 Intrinsic failure rate improvement trends for IBM technology. (a) Logic circuit reliability, where the per-circuit percent failure rate is given per 10³ hours. The numbers noted on the curve refer to the number of circuits per chip. (b) Bipolar memory reliability, where the per-bit percent failure rate is given per 10³ hours and the numbers noted on the line refer to the number of bits per chip (K = 1024).

in question. This system was to be called Remote Technical Access Information Network (RETAIN).

At the same time that the planning and development work on RETAIN had been progressing, experimentation with technical instruction via the computer had also been underway. It offered advantages in rapid course update and evaluation. This project was to be called Computer Assisted Instruction (CAI). In 1965, both of these projects were initiated in selected locations on an experimental basis. By September 1967, the two applications were integrated under a single software system and supported 112 terminals.

The use of a remote data bank increased in importance as a maintenance aid. A test center for teleprocessing products was brought on line in 1969. This center was developed to provide the CE with a fast, efficient means of testing and verifying the performance of teleprocessing products by providing an alternate host capability. This allowed "off-line" maintenance of teleprocessing equipment without interruption of other data processing operations.

Two important improvements were made available in 1970, in a new version of RETAIN. A capability for structuring a search argument for a particular product symptom was developed to provide more efficient and rapid data location. This capability reduced the teleprocessing data loading for each problem inquiry. The second improvement was the provision for a data link between the customer's system and the RETAIN system. This data link would now allow a specialist at a remote location to operate the customer's system while utilizing the full library of diagnostic tests available at the customer's site.

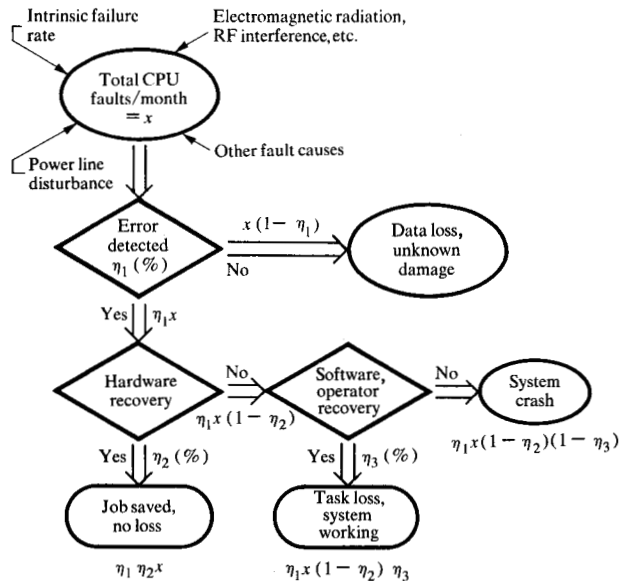


Figure 3 CPU recovery from faults.

Results of the diagnostic runs could be displayed on a terminal for analysis by the specialist, with the CE maintaining telephone contact to assist in solving the problem. By 1976, there were a series of on-line data systems in various locations around the world providing RETAIN services for our CEs on a worldwide basis. In the following year, a worldwide data-link capability was provided.

Progress in CPU and system RAS techniques

• System operation in the presence of faults

The basic treatment of faults is shown in Fig. 3. In a computer system, faults are caused by the intrinsic hardware failure rate, by external influences, by design errors (timing, circuit input pattern sensitivity, circuit overloading, mistakes in logic, programs, etc.) and by operator mistakes. The first obvious step for good RAS is to reduce the occurrence of faults. If the fault causes an error, the first step in recovery is detection of the error. If the error is not detected, computer system operation continues and the erroneous results propagate. Information is destroyed or modified erroneously and must be corrected at some time in the future.

When an error is detected, the procedure is to perform information-damage assessment, counteract the effects of the fault, isolate it, and treat it properly. This treatment may range from ignoring the fault to immediate repair. Finally, system recovery must be effected and system operation resumed. As technologies and the use of computer systems changed, the necessary technical in-

novations needed for successful operation in the presence of faults were implemented in IBM systems.

• Very early computer systems (early 1950s)

In very early systems, each IBM installation had CEs either resident or on quick call. Error detection was normally delayed, and the CEs isolated system faults through re-creation of the failure. The CE then used the logic drawings of the system, detailed electrical diagrams of the pluggable units, simple diagnostic programs, and the console and an oscilloscope to complete his analysis [6].

At this time, the concept of using the computer to test itself and help locate faults was born. The idea of having specialists prepare a general set of tests to try to ensure the absence of faults (test routines) and to locate faults (diagnostics) was proposed and implemented [6]. However, re-creating the failure symptoms for a wide variety of cases proved to be unexpectedly difficult. The procedures were temporarily successful because of the ability and dedication of the CEs, but better technology was clearly needed. Error detection was needed since considerable computer time was wasted using invalid data and diagnosing computer faults. The first diagnostic routines, while making an improvement, were not adequate because error re-creation using software was difficult. The idea of backward error recovery, specifically *checkpoint-restart*, was invented early on. Enough information to restart the program was stored on an external medium and, after an error was discovered, the program could be resumed at such a point. However, this also was satisfactory only for the less complex programs and systems.

• Early computers for defense

Early computers designed for defense applications required good RAS and, because they were constructed from relatively unreliable components, were carefully self-checked. An impressive example is the AN/FSQ-7 (SAGE) computer [7], used for real-time processing of radar scans for air defense as well as for potential missile launching. It was designed with fault-tolerant features to achieve high availability and accuracy. The central processor (still using vacuum tubes) contained one operating computer and a second acting as backup. Each computer had elaborate fault-detection schemes using parity and software testing and diagnostic programs. The I/O had standby spares. The standby computer executed self-tests frequently and its memory was periodically updated with the current state of the operating computer. After error detection, the switchover and recovery were executed by software. This effort showed that reliable operation could be attained, although the cost was high; and much valuable experience resulting in improved RAS technology was gained. Error detection was increased,

Table 1 RAS features of some early IBM computers.

Machines	Checking schemes					
	Data flow paths	Control units	Arith. units	Memory units	I/O units	Misc.
IBM 650	1. 2/5 code (single error detection)	1. clocking 2. proper sequence of control signals 3. duplicate circuitry 4. accumulator interlock	1. bi-quinary codes 2. sign-agreement checking 3. correct complement add and true add	1. 2/5 codes (SED)	1. card reading 2. card punch 3. tape system parity check (BCD)	Program checks include: 1. invalid operation code 2. invalid addresses 3. overflow 4. branch distributor codes check 5. interlocks
IBM 7070	1. 2/5 code (validity check)	1. validity checks 2. clocking	1. 2/5 code 2. sign-agreement checking	1. address checks 2. data in and out are validity-checked	1. BCD one-bit parity check 2. card reading and punching 3. two-gap head, dual-level sensing in tape system	Program checks include: 1. invalid operation code 2. invalid addresses 3. use of instruction counter for error routine
IBM 7030 (Stretch)	1. parity check	1. validity checks 2. clocking 3. parity checks on various register fields	1. modulo-3 residue check for floating point arithmetic 2. parity	1. modified Hamming code for single error correction and double error detection	1. 720 tape, bit error detection 2. 7302 disk usage SEC-DED code	1. some duplication in circuitry

redundant information was stored routinely for recovery, computer self-tests and diagnostics were improved, and the resulting serviceability was improved.

• Batch data processing

The first stored-program computers introduced by IBM in the field of commercial computer processing, the 701 and 702 systems, used vacuum tubes as the active devices and germanium diodes for logic switching. Field-replaceable units for these machines were made up of eight vacuum tubes and associated passive components. These plug-gable packages aided in fault diagnosis since they were larger than previous pluggable packages.

When the 650 was introduced, more RAS features were included in its design (see Table 1). In addition, the 701/702 evolved to the 704/705 with floating-point arithmetic (and thus a more complex structure). To cope with this, extended CE education was begun. The first general RAS support attempted was the production of self-test routines and diagnostics to aid in re-creating the effect of errors in controlled situations, and to facilitate the logical analysis

necessary to locate the fault that caused the error. These programs applied sequences of input vectors to logic circuits by loading registers accessible to the program and then applying standard computer instructions.

The early IBM work for AN/FSQ-7 emphasized the necessity for good diagnostics [8]. To test for possible faults the "start large" approach was used. The system was stressed as much as possible and difficult CPU operations were run concurrently. The idea was to obtain maximum fault coverage as quickly as possible. If a fault were indicated or if diagnostics were being run because of known difficulties, the "start small" approach was used. The simplest operations were performed and then tests using small incremental amounts of circuitry were run. Specialized routines were available for suspected units. These efforts were supported by increased CE educational efforts.

However, analysis indicated that software tests have serious deficiencies [9]. The first is control of the elements being tested. The fraction of circuits accessible to

direct control is small, so writing a software program to produce a desired test pattern is difficult. The second difficulty is the state-resolution problem: the interrogation of the results of the test-pattern application. Several instructions and many intermediate states and timing cycles have to be used. Finally, test coverage is unmeasurable, and control of the system of programs is difficult.

The 702 had parity checking in its memory [10]. The 705 also had parity checking for each character. This improved data integrity and also helped RAS because of the additional assurance such error detection gave to commercial data processing, which differs from scientific processing in that checking without duplication is difficult. These systems were followed by the 7090 and 7094, which benefited from improved components (transistors and magnetic cores) first introduced in the 7030 system (described next).

- *Unique systems*

The 7030 (Stretch) computer [11] used two innovations that have had a long-range impact on RAS. First, it used *single-error-correction, double-error-detection* (SECDED) codes for main storage, and parity, duplication, and modulo-3 codes for error detection. These characteristics allowed a single error in the memory to be automatically corrected and the fault to be located and later removed during a scheduled maintenance period. Another benefit was the ability to provide fault isolation, since the error-correction process requires identification of the failing bit. The second innovation was the storing on punched cards of the status of all processor latches immediately following the detection of an error. Table 1 compares various checking/correction techniques implemented in the IBM 650, 7070, and 7030 systems.

The 7030 computer was one of the earliest systems to use the *standard modular system* (SMS) cards, which were the field-replaceable units (FRUs). The 7030 also used transistors and magnetic cores as its logic and storage components, which provided a great advantage over tubes and electrostatic storage. The circuit logic was shown by printed logic drawings, produced by IBM's first design automation system. These drawings, one page per FRU, aided fault determination by error analysis.

The 7030 had these RAS innovations because it was the most complex (as well as the fastest) computer system built up to that time by IBM, and new concepts were needed to achieve the required reliability. Error detection made error analysis much easier, although there were still difficulties with producing good diagnostics. Re-creating the environment that produced the error from the fault was aided by the logout; however, this was awkward in its early form.

The SABRE system (American Airlines Electronic Reservations System) [12] did the first real-time processing for airline reservations using two IBM 7090s in on-line and standby roles supporting a network of terminals and communications processors. In order to make this system successful, not only was there redundant equipment, as in SAGE, but much work was done on improving the techniques available for the recovery programs.

- *System/360*

With improved capabilities emerging in both hardware and software, companies began to use computers interactively in their day-to-day affairs. At this point, it was becoming clear that in computer system RAS design primary emphasis should be placed on the error-detection capability of the system, since without this first step recovery is greatly hindered.

Error detection is required for all types of errors, whether caused by solid or intermittent faults, so that when an error is detected, a recovery technique can be invoked to ensure data integrity and availability of a valid system. Furthermore, when hardware checkers for error detection are designed and placed at a suitable location, the error indication will provide maximum capability for fault isolation, so that the failing FRU can be identified quickly and service can be accomplished in a minimum time. System/360 implemented many novel checking circuits [13], such as parity-predict adders, carry-dependent-sum adders, and two-rail logic checking.

Another new idea was to reduce diagnostic time and improve the precision of fault location by using *fault-locating tests* [14]. These tests used hardware to implement a process called *scan-in, scan-out*. Scan-in, scan-out allowed testing of combinational circuits by adding the hardware necessary to read a pattern from tape, insert the bits in latches, step the computer one or more cycles, then examine an output bit. Test patterns were generated by computer from circuit diagrams, using the single-stuck-fault model, and patterns sequenced using sequential testing techniques. One combinational logic step was satisfactory for the CPU, but the sequential nature of the channels required several steps—*sequential scan* [15].

These tests were controlled by the microprograms, and this led to the idea of microdiagnostics [16]; *i.e.*, since the control and interrogation in a few cycles were done by the microprogram, the diagnostic programs were put in ROM [14]. This improved flexibility and allowed very good tests to be written for storage in System 360/50 as well as automatic testing of the System 360/40 CPU whenever the console START button was pressed. The System 360/30

used microdiagnostics for circuit-level tests and for tests to assist debugging. The System 360/25 was the first IBM CPU with loadable control store; this eliminated the storage space constraint for microdiagnostics.

System/360 computers were designed so that the logged-out information about CPU latch status was stored in main storage. This could then be analyzed by software routines and the pertinent information preserved or printed for the customer engineer. This idea was improved and joined with wide use of microdiagnostics for the later models of System/360 [17, 18].

A standard diagnostic monitor was written to control the software programs written at many locations, and interfacing with the standard supervision program was begun so that maintenance began to be integrated with the rest of the system functions. Special diagnostic hardware—"channel wraparound" and storage, channel and I/O control unit state latches accessible to CPU interrogation—was added [14].

Maintenance analysis procedures (MAPs) were introduced to aid the CE in his diagnostic task. These MAPs provided a step-by-step process to isolate the cause of a failure and tended to reduce the need to teach CEs how a product worked, permitting emphasis of a "how-to-fix" maintenance philosophy [19]. MAPs were first applied to I/O products and small systems.

With the advent of System/360 and its supervisor program (OS), IBM was trying to do the whole difficult job of recovery and multiprocessing for the first time. Based on practical experience, the seminal papers on recovery were written early [20, 21]. Ideas described in those papers have been expanded and incorporated into standard IBM products [22-24].

System/370

The ideas used in System/360 were continued and improved with the introduction of System/370; the concept of a unified hardware/software system for RAS, as shown in Fig. 1 in the introductory section, has become a reality. A hierarchy of features now exist to aid recovery from intermittent and solid faults; these features are implemented in microcode, hardware, and software and they reside in the box, subsystem, and system levels.

- *Error-correction codes for main memory*

In 1968, a new class of single-error-correction, double-error-detection codes was invented by two IBM engineers [25]. These were called *odd-weight-column* codes [26]. With the same coding efficiency, this new class of codes made improvements over the standard Hamming

code implementation in cost, performance, and reliability. Examples of these new codes were implemented in the IBM System 370/158 and 168, and later in the 3031, 3032, 3033, and 4300 series as well as in many other IBM systems. Since its first publication in 1970 [26], this class of codes has been widely used in the U.S. and abroad.

- *Hardware retry*

CPU instruction retry was first commercially implemented on the System 370/155 and 165 [17], and followed by the System 370/145. The techniques of checkpoint-restart were designed for a single instruction to overcome the effect of an intermittent error in the CPU. Data used by the instruction are stored at appropriate points during instruction execution, and the instruction progress is charted by a set of states. When an error is detected, the microcode interrogates the state to determine the last valid data, restores the operands from these data, and begins from the last valid state. A count is kept in case the error is caused by a permanent fault, in which case, when the count reaches a predetermined value, the retry is terminated and a permanent fault is signaled. Since the basic IBM patents in 1968 [27-29], the computer industry has widely implemented the instruction-retry mechanism.

A method of retrying catastrophic channel-check errors by OS/360 software augmented by hardware was implemented on the System 370/155 in 1971. A hardware and microcode *channel/control unit retry* was also introduced on the 370/155. When errors in a particular class, e.g., during the Direct Access Storage Device (DASD) SEEK time, were detected by a control unit, a unique ending response was sent to the channel. The channel recognized this and reissued the last channel command. This retry was transparent to the software since no interrupt was issued at the program level.

Studies in the early System/360 design revealed a number of occasions when a second channel interface or a DASD controller would allow both channel overloading and channel interface errors to be bypassed; therefore, in 1971 *I/O path switching* was introduced on the 2314 DASD family. Previous design had allowed sharing between processors; this was now applied to a single processor with multiple program-switchable I/O paths [30, 31].

Such abilities are part of the standard MVS supervisor [32], and the improved availability MVS offers stems mainly from its ability to automatically reconfigure hardware components.

- *New diagnosis and testing techniques*

For effective self-repair features, all faults must be efficiently located. In the early 1960s, K. Maling, M. Evans,

and others under R. J. Preiss' direction wrote a test generator and deductive fault simulator program that was used to generate test patterns for the larger models of System/360 [15]. These programs were heuristic and not completely satisfactory. In 1966, Roth published a basic paper showing a new technique called the *D-calculus* for generating test patterns [33]. The importance of this technique was quickly recognized [34] and has been widely used to find test patterns for combinational circuits since. Another technique, using the *Boolean difference* [35] for test-pattern generation, has been an important tool for analyzing testing theory.

It was with System/370 that the idea of an *autonomous diagnostic processor* was begun as a supplement to the CE control. A universal system service adapter was incorporated in the System/370 Model 155 which provided a standard interface to external equipment for testing the 155 when it was in a stopped or disabled condition [36]. This idea was extended to an independent processor which could control execution of diagnostic routines as the CE did at the console. Program trace facilities, the ability to capture logout data, continuous monitoring of selected logic points, and programs to analyze the captured data have been added.

MVS software supervisor

Multiprocessing (MP) is one means of increasing availability; another is eliminating the need for unscheduled shutdowns. When an error occurred in previous systems, the system could not do any work until the installation reinitialized the system. One way of carrying out these procedures is by using a software supervisor which makes good use of the available hardware features. The standard IBM supervisor for System/370 is MVS; when an error occurs in the MVS system, the system attempts to continue operating. MVS attempts to retain availability through error-recovery routines that isolate the record, clean up and repair, and retry and reconfigure. Processing continues while the system carries out these tasks. Primarily recovery management support and the recovery termination manager perform these functions.

The improved availability MVS offers derives from the ability to automatically switch from a failing unit to an alternate. In addition, it is possible to reconfigure hardware components to fit an installation's needs or to reconfigure hardware components allowing service personnel to perform concurrent maintenance. Thus, over a period of time the system does more work because it loses less time due to failing hardware. A multiprocessor installation can be divided into two systems so that only the hardware components actually required for the special system are allocated to one processor, leaving the bal-

ance of the hardware resources available for normal work on the other processor. Thus, MP not only does more work in the sense of doing two things at one time, but also is more available, responding to the different needs of an installation at different times.

New RAS features in recent IBM systems

New features to improve RAS have added innovations in the IBM 4300 systems. For example, the difficulties of test generation are made more acute by large-scale integration (LSI) with the large number of circuits that can now be placed on a single chip, and the impossibility of probing to find faults. A method of overcoming these difficulties makes use of *level-sensitive scan design* (LSSD) [37]. In this method, all latches in a chip are connected in a shift register as well as with their normal interconnections. The shift registers can easily be tested, and the other circuits are tested as combinational circuits.

As discussed previously, instantaneous *error detection* and efficient *failure isolation* (ED/FI) are essential to system RAS. In order to obtain a quantitative measure, a basic evaluation method of a system's ED/FI capability is required as the system is being designed, so that efficient ED/FI capability can be achieved. Beginning in 1972, an ED/FI evaluation technique was developed in IBM, was tested, and now is widely used while IBM products are being designed [38]. With ED/FI, the fault model is defined in terms of failure probabilities associated with checkers and syndromes. Failure likelihood and probability of error detection are calculated from the circuit count, failure rates, and check placement. This technique is especially significant for LSI designs; it is extremely important to have error-detection and fault-isolation capability built into the early logic design phase. The evaluation and design ideas go hand in hand with a maintenance philosophy which relies primarily on instantaneous error detection and isolation of faults in the operating environment, rather than on conventional diagnostic methods of error re-creation. The result of the ED/FI design evaluation determines the capability of the system for diagnostic problem determination, remote diagnostics, and customer service.

This approach led to designing some IBM systems so that service tasks normally executed by CEs can be performed by the customer's operators. Built-in error-detection and isolation circuitry isolate the failure automatically or semi-automatically, and the customer is instructed to run additional tests by activating diagnostics housed in the unit. Test results point the customer to the correct place in the documentation, which instructs the operator as to the corrective action. Customer-replaceable elements are easily accessible and are also guided,

keyed, and color-coded to ensure correct positioning. The 3101 terminal announced in 1979 was the first IBM product to incorporate CPAR (Customer Problem Analysis Resolution).

To reduce costs, eliminate delay, and allow independent customer ability to set up or relocate a product, starting with the IBM 3767 in 1975, a practice called *customer set-up* (CSU), whereby a customer operator can perform all of the procedures required to get a product into operation on his system without IBM assistance, was initiated. The final test of a product, after CSU is complete, is activated by a simple key action which causes an automatic repeat of the final test performed in manufacturing prior to shipment. This concept has now been implemented on the IBM 3270 display system, including the IBM 3287 and 3289 printers in 1978, the IBM 8130 and 8140 systems in 1979, and many other display/printer products.

In addition, portable service processors are available for special system problem determination, channel monitoring, and line/link monitoring. A portable service processor (the IBM Maintenance Device) was first introduced in 1979 with the IBM 8130, 8140, 3370, and 3380. It includes a microprocessor, file, keyboard/display, communications port for remote support, and various ports to connect to a variety of products. A diskette containing a unique maintenance package, including *MAPs-Diagnostic Integrated* (MDI) error analysis programs and symptom/failure indexes, is shipped with each product and is loaded at the start of a CE call.

The next step was to use a communications network for remote control and diagnosis of the ailing computer [16] so that the logout patterns of a good computer can be compared with those of a failing computer. Patterns making faults appear as errors are captured and used for remote diagnosis and remotely controlled testing hardware.

These extended facilities improved hardware RAS sufficiently so that the main difficulties arose from the more complicated software which was attempting to make good use of the expanded hardware functionality. Early programs were so small that they could be easily comprehended and fixed, but with the advent of higher-level languages, large main storage, and multiprogramming, software errors have increased. IBM effort in software RAS has been concentrated in three fields: conventional fault avoidance, support for hardware features, and basic recovery with operating systems and the problem of large data base recovery. The major RAS enhancements of the last twenty-five years are exemplified in Fig. 4.

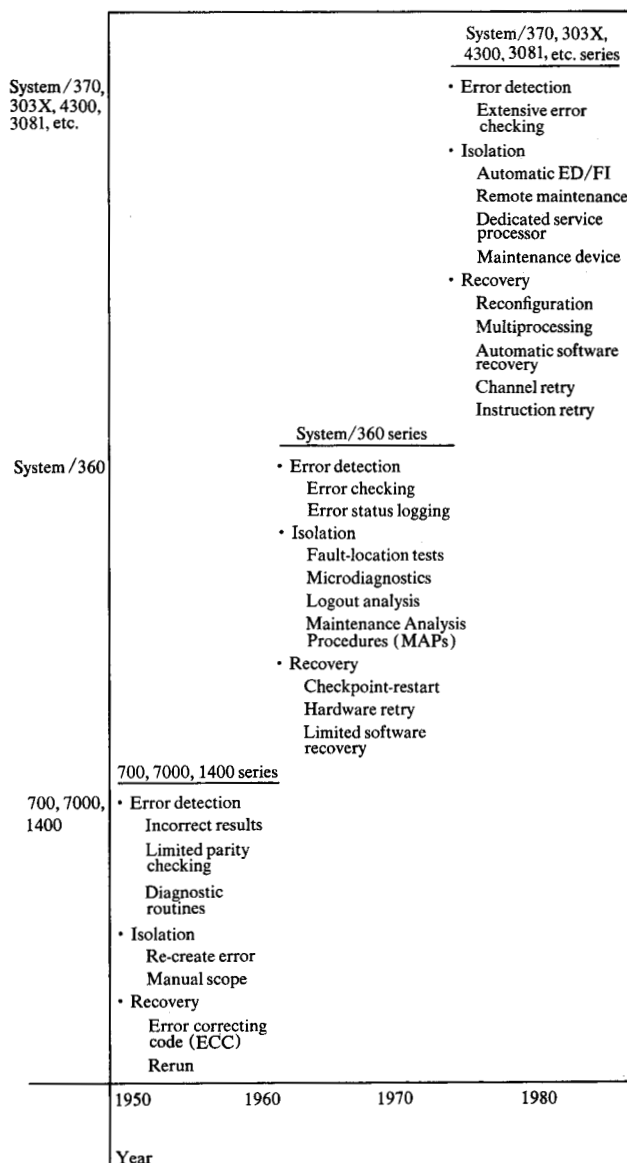


Figure 4 RAS enhancements.

RAS progress in magnetic storage devices

• Magnetic tape area

Since the early 1950s, IBM has made very significant progress in magnetic tape technology. The paper by Harris *et al.* [39] in this issue has more detailed discussion on this subject. In this section, we shall only highlight the RAS progress and assess its impact on the overall technology progress.

To assess the impact of RAS progress on tape technology, we shall use areal density as a key parameter to measure progress. As areal density increases, the cost per

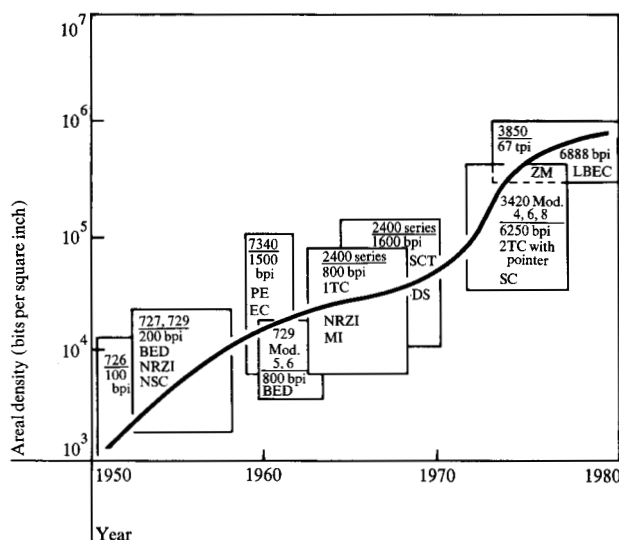


Figure 5 Progress in IBM tape technology. The following abbreviations have been used: BED = bit error detection, NSC = non-self-clocking, SC = self-clocking, PE = phase encoding, EC = erasure correction, 1 (2) TC = 1-(2-) track correction, MI = mechanical improvements, SCT = self-clocking tracks, DS = de-skew, LBEC = long-burst error correction.

bit goes down and total storage size and data rate increase, which means a better figure of merit to end users. As shown in Fig. 5, the 726 was the first IBM parallel track tape unit to use the *nonreturn-to-zero inverted* (NRZI) code, which is 100% efficient in recording density. (Here the efficiency is defined as the data bits divided by flux reversals within a unit recording space.) The key problem in the NRZI code is that it does not have a self-clocking capability; when bits from different tracks arrive at a variable time due to mechanical and electrical skew, an error occurs. Therefore, improvements were made to reduce the mechanical tolerance in the tape transport by tuning the delay line or buffer register to the read/write process. The NRZI technique reached its limit in the 1960s when the linear density in the IBM 729 Models 5 and 6 and the early models of the 2400 tape series was up to 800 bits per inch (bpi).

Meanwhile, the capability of error-correcting codes, matched with the NRZI code, was greatly enhanced. Both the 727 and 729 tape systems had a *vertical redundancy check* (VRC) and a *longitudinal redundancy check* (LRC) for error detection. Besides error-detection codes in the half-inch tape system, it had read-after-write check to make sure the recorded data were correct. As the density increased, a stronger correction was needed. In the 2400 tape series, a combination of *cyclic redundancy check* (CRC) and VRC made a clever single-track correction system [40, 41] which can correct a total track failure within a block of data in the nine-parallel-track system.

This contribution by IBM was a major step beyond the state of the art and later became the industry standard.

Before the 2400 tape system, the IBM 7340 tape system used the phase-encoding (PE) technique for data encoding; this technique has a lower efficiency in recording density than the NRZI code for representing a data bit. However, the PE code has a self-clocking feature, and its dual-level sensing provides a binary erasure channel property which, by using a (10,8) code [42], can correct all single errors and 33 out of 45 double errors within a codeword. IBM soon went back to the standard nine-track system because the ten tracks for the 7340 were non-conventional. No following systems have used ten tracks.

The PE data encoding method was used, however, in the 2400 tape series because of its self-clocking property, which improved the linear density up to 1600 bpi. The erasure-channel correction was not used, but the single-track correction system was employed. The PE code was used until the introduction in 1973 of the 3420 tape series Models IV, VI, and VIII, which used a (4,5) run-length code [43] for data encoding. This code has better (69% vs. 50%) recording density efficiency than the PE code. For these models a new code called *group-coded recording* (GCR), with a more powerful error-correction system, was also invented by IBM engineers [44, 45]. Significant progress has been made with this code, pushing the linear density from 1600 to 6250 bpi. The GCR code became the U.S. industry standard as well.

GCR enables the nine-track system to correct any two-track failure with a real-time pointer [46] and any single-track failure without a pointer. The GCR system has also helped move the data rate up to 1.25 megabytes per second. The error-recovery technique of the tape control unit also provides error-detection and re-read capability to take care of intermittent errors during the read process. In addition to the GCR code, there is a two-8-bit CRC at the end of each record for extra detection capability [47].

Along with the coding progress, improvements made in tape oxides and substrates, motion controls, direct-drive dc motors, and solid state electronic technology, improved contour-on-head design, and more electronic buffering to trade off with mechanical tolerance have pushed tape technology up to 9042 flux changes per inch, i.e., 6250 bpi. While progress was being made in the half-inch tape area, IBM devoted great resource to develop a *mass storage system* (MSS), in which the data storage is even greater in size with extremely low-cost data, for replacing the half-inch tape library. Because the IBM MSS 3850 uses a cartridge and a rotating head concept, a new data encoding method without a dc component was required. This requirement led to the invention of *zero modulation*

(ZM) code. The cartridge concept provides a different data format from the conventional parallel-track system, thus abandoning the track correction scheme. Instead, it uses an interleaved subfield code to correct a single burst of 128-bit-long or nonoverlapped 8-bit burst in 16 different sections [48]. The ZM code not only has a self-clocking feature and 100% recording efficiency, but also provides a powerful pointer to pick up extra error-detection capability in addition to what ECC provides.

• Magnetic disk area

Since the introduction of the RAMAC 305, IBM disk technology has made great progress in density, storage capacity, data rate, cost, and reliability [49]. This section will highlight only the RAS aspects of IBM disk technology. Using areal density as a figure of merit to discuss the progress, Fig. 6 shows that we have made an improvement of about four orders of magnitude *i.e.*, from 10^3 to 10^7 bits per square inch (bps).

As compared with magnetic tapes, the disk is of a better, hence more costly and more rigid, material and is a more precisely controlled device; it is therefore inherently more reliable. Because it is a serial-access device with respect to one track at a time, a skew problem of bits in multi-track arrival at different times does not exist. The disk has a higher data rate due to its rotation speed than tape. Its sensing circuit design has to allow greater tolerance, and hence the clocking window for sending data is narrower. The data encoding requirements for disk are not completely the same as for tape. In the early low-to-medium-density systems, the data-encoding method used was mainly to achieve specified density. In the 1301 and 1311 disks, the NRZI code was used for data encoding. As the areal density increased, a code called "Double Frequency" was used for data encoding in the 2302, 2311, and 2314. This code has a structure similar to that of the phase-encoding method; it is simply to insert a clocking pulse in between ones. In the very early disks, *i.e.*, RAMAC 305 I and II, only simple parity checks were used for checking data integrity. The IBM 1301 was the first commercially available mass-produced disk using a 16-bit polynomial code for improved error-detection capability. This burst-error-detecting code was also used in the follow-on products, making significant progress in moving away from parity check only.

With the introduction of the IBM high-density and high-data-rate disk (3330) in 1971, basic disk technology made significant progress. A different data encoding method called *modified frequency modulation* (MFM) was used. This code provided a synchronized capability without requiring insertion of extra bits as in the double-frequency code. It also had a better capability to handle

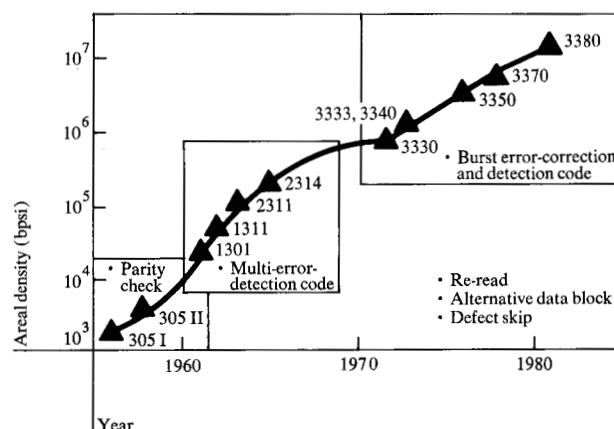


Figure 6 Progress in IBM disk technology.

off-track interference. After nearly eight years in the burst-error-detecting mode, the large disk commercial product made a great leap forward in reliability improvement. The IBM 3330 used a Fire code [50] to correct a burst error up to 11 bits and had a very high error-detection capability up to a burst length of 45 bits. Besides error-correcting codes, this device could perform alternative track selection for avoiding a bad track and an echo check to make sure data had been written on the disk.

The 3330 disk technology and its RAS features set a milestone in the I/O industry. Soon its features were widely adopted within the U.S. and abroad. As the disk areal density moved upward in the 1970s, MFM continuously performed well in data-encoding requirements. Error-correcting codes were used basically as the Fire code but varied in burst-correction capability. In 1979 the IBM 3370 disk design used interleaved b-adjacent code [51] to replace Fire code for error correction.

Besides ECC, other important recovery techniques used were *defect skip*, *alternative data block*, and *re-read* in the IBM disk product family to enhance reliability and data integrity. It should be pointed out that significant basic technological progress has been made to improve device reliability in spite of increased areal density.

Additionally, in the early 1960s IBM was the first company to implement the parallel linear-feedback shift-register scheme in the 2820 drum control unit. Since the first disclosure of the theory behind it [52], industry has applied it widely in parallel CRC implementation for tape and for communication networks as well.

The improvements in recording-media quality and servo and head designs, and progress in error-correcting codes, error-recovery techniques, and data-encoding methods have made the high-density tape/disk product

even more reliable than before. In summary, the improvement in areal density of magnetic recording devices has brought great improvement in cost per bit, data rate, and total on-line storage capacity.

Conclusions

In spite of increased product and system complexity, advances in reliability, availability, and serviceability have made it possible for users to place additional reliance on computers, so much so that many users have committed much of their business to data processing systems. This has been possible because of the innovations described in this and other articles in this issue.

The beginning was made with improvements in the area of error detection, which enables dynamic fault isolation and is the basis for recovery and data integrity. Additionally, technology has produced significant gains in reliability. This has been augmented by the use of error-correction codes and redundancy. Availability is, in part, achieved through a recovery hierarchy that forces recovery at the lowest possible level. Serviceability has been significantly enhanced with automatic real-time fault isolation and, when necessary, remote assistance.

The future holds many challenges and the demand for system integrity will continue to increase, which dictates the need to continue to improve the state of the art in the field of RAS.

Acknowledgments

The authors thank E. C. Byman for his support of this paper, as well as P. E. Barshinger and R. C. Williams for their assistance, and G. R. Santana for his comments on the disk section. We would also like to thank all of the many IBMers who have contributed to the realization of the actual progress in IBM RAS technology.

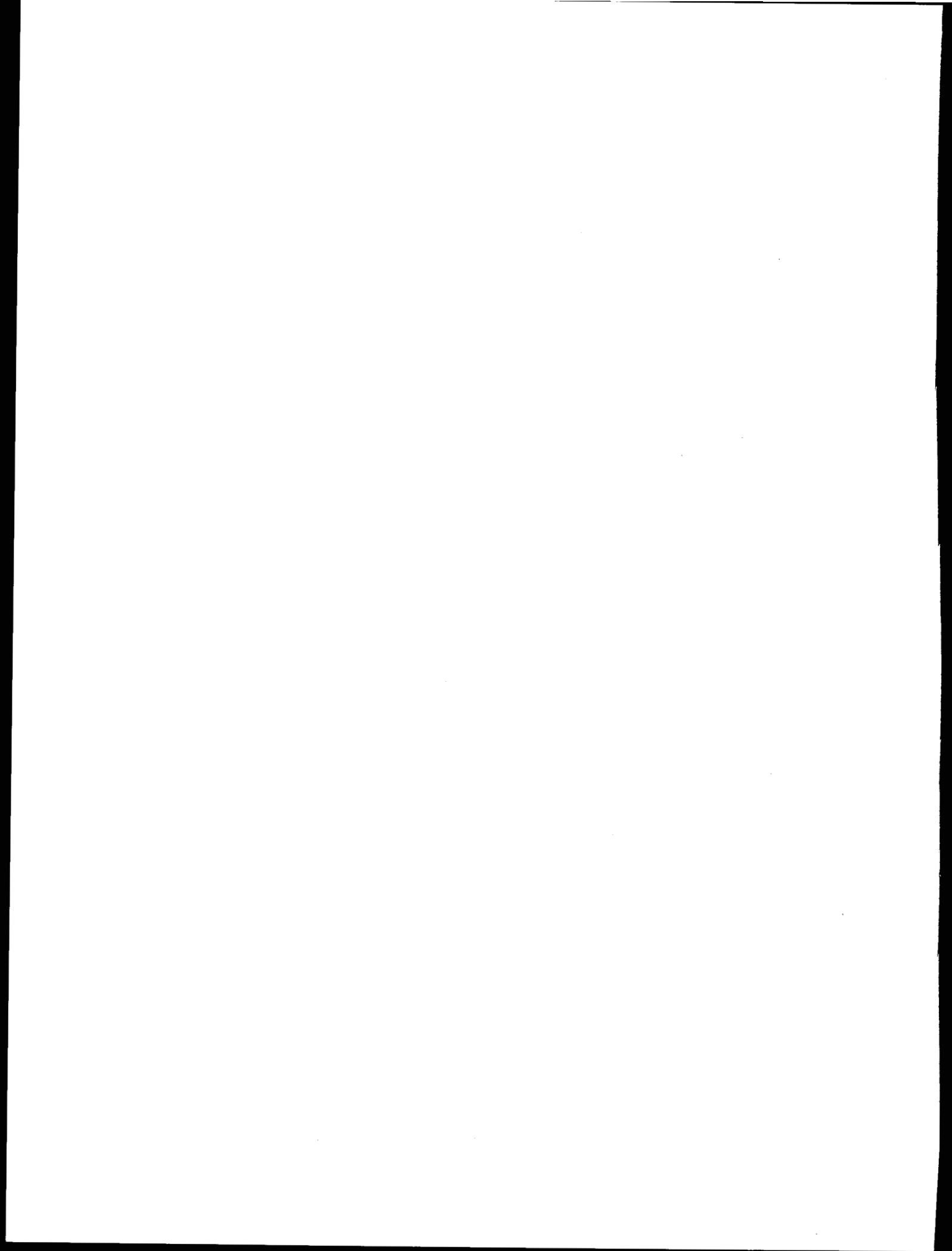
References

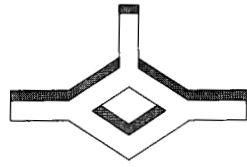
1. *Computing Systems Reliability, an Advanced Course*, T. Anderson and B. Randell, Eds., Cambridge University Press, Cambridge, England, 1979.
2. J. von Neumann, "Probabilistic Logic and the Synthesis of Reliable Organisms from Unreliable Components," *Automata Studies*, C. E. Shannon and J. McCarthy, Eds., Princeton University Press, Princeton, NJ, 1956, pp. 43-98.
3. S. E. James, "Evolution of Real-Time Computer Systems for Manned Spaceflight," *IBM J. Res. Develop.* **25**, 417-428 (1981, this issue).
4. P. F. Olsen and R. J. Orrange, "Real-Time Systems for Federal Applications: A Review of Significant Technological Developments," *IBM J. Res. Develop.* **25**, 405-416 (1981, this issue).
5. David R. Jarema and Edward H. Sussenguth, "IBM Data Communications: A Quarter Century of Evolution and Progress," *IBM J. Res. Develop.* **25**, 391-404 (1981, this issue).
6. L. R. Walters, "Diagnostics Programming Techniques for the IBM Type 701 E.D.P.M.," *Convention Records of the IRE, 1953 National Convention*, New York, NY.
7. R. R. Everett, C. A. Zraket, and H. D. Benington, "SAGE—A Data-Processing System for Air Defense," *Proceedings of the Eastern Joint Computer Conference (EJCC)*, Washington, DC, 1957, p. 148.
8. J. J. Dent, "Diagnostic Engineering Requirements," *AFIPS Conference Proceedings* **32** (1968 Spring Joint Computer Conference, Atlantic City), 503-507 (1968).
9. M. Ball and F. Hardie, "Effects and Detection of Intermittent Failures in Digital Systems," *AFIPS Conference Proceedings* **35** (1969 Fall Joint Computer Conference, Las Vegas), 329-335 (1969).
10. C. J. Bashe, P. W. Jackson, H. A. Mussell, and W. D. Winger, "The Design of the IBM Type 702 System," paper no. 55-719, *AIEE Transactions* **74** (Part I, Communication and Electronics), 695-704 (1956).
11. W. Buchholz, Ed., *Planning a Computer System (Project Stretch)*, McGraw-Hill Book Co., Inc., New York, 1962.
12. M. N. Perry and W. P. Plugge, "American Airlines 'SABRE' Electronic Reservations System," *Proceedings of the Western Joint Computer Conference*, Los Angeles, 1961, pp. 593-601.
13. F. F. Sellers, M. Y. Hsiao, and L. W. Bearnson, *Error Detecting Logic for Digital Computers*, McGraw-Hill Book Co., Inc., New York, 1968.
14. W. C. Carter, H. C. Montgomery, R. J. Preiss, and H. J. Reinheimer, "Design of Serviceability Features for the IBM System/360," *IBM J. Res. Develop.* **8**, 115-126 (1964).
15. R. J. Preiss, Chapter 7, *Design Automation of Digital Systems*, Vol. 1, M. A. Breuer, Ed., Prentice-Hall, Inc., Englewood Cliffs, NJ, 1972, pp. 335-410.
16. F. J. Hackl and R. W. Shirk, "An Integrated Approach to Automated Computer Maintenance," *IEEE Conference Record on Switching Theory and Logical Design* **16-C-13**, 289-302 (1965). See also A. M. Johnson, Jr., "The Microdiagnostics for the IBM System 360/Model 30," *IEEE Trans. Computers* **C-20**, 798-803 (1971).
17. J. Fox, "Availability Design of the System/370 Model 168 Multiprocessor," *Second USA-Japan Computer Conference Proceedings*, Tokyo, August 1975, pp. 52-57.
18. A. N. Higgins, "Error Recovery through Programming," *AFIPS Conference Proceedings* **33** (1968 Fall Joint Computer Conference, San Francisco), 39-43 (1968).
19. D. C. Burnstine and W. H. Eppard, "Maintenance Strategy Diagramming Techniques," *Proceedings of 1966 Annual Symposium on Reliability*, San Francisco, pp. 497-506.
20. L. A. Bjork, "Recovery Scenario for a DB/DC System," *Proceedings of the ACM Annual Conference*, Atlanta, 1973, pp. 142-146.
21. C. T. Davies, "Recovery Semantics for a DB/DC System," *op. cit.*, Ref. 20, pp. 136-141.
22. *Fast Path Feature General Information Manual*, Order No. GH20-9069-1 (1976), available through IBM branch offices.
23. *Information Management System/Virtual Storage (IMS/VS), System Programming Reference Manual*, Order No. SH20-9027-2 (1975), available through IBM branch offices.
24. *OSVS2 MVS Overview*, Order No. GC20-0954-0, available through IBM branch offices.
25. M. Y. Hsiao and E. Kolankowsky, "Optimum Apparatus and Method for Check Bits Generation and Error Detection, Location, and Correction," U.S. Patent 3,623,155, November 23, 1971.
26. M. Y. Hsiao, "A Class of Optimal Minimum Odd-weight-column SEC-DED Codes," *IBM J. Res. Develop.* **14**, 395-401 (1970).
27. M. Bee, D. J. Lang, and A. D. Snyder, "Data Processing Machine Function Indicator," U.S. Patent 3,539,996, January 15, 1966.
28. M. Bee and D. J. Lang, "Instruction Retry Byte Counter," U.S. Patent 3,564,506, January 12, 1968.
29. B. McGilvray, D. J. Lang, W. E. Bochner, and M. W. Bee, "Data Processing System—Execution Retry Control," U.S. Patent 27,485, January 15, 1968.

30. R. A. Bell, "(An) I/O Switching (Scheme) for Multiprocessors," *Technical Report TR00.2105* (1970); available from IBM Data Systems Division laboratory, Poughkeepsie, NY.
31. J. F. Thompson and C. A. Zito, "Channel Status Checking and Switching System," U.S. Patent 3,286,240, March 3, 1971; W. Clark, K. A. Salmond, and T. S. Stafford, "Input/Output Control," U.S. Patent 3,725,864, March 3, 1971; E. W. Devore, R. J. Smith, and J. M. Tyrrell, "Input/Output Unit Switch," U.S. Patent 3,372,378, March 5, 1968.
32. *MVS Diagnostic Techniques*, Order No. GC25-0725-2, available through IBM branch offices.
33. J. P. Roth, "Diagnosis of Automata Failures: A Calculus and a Method," *IBM J. Res. Develop.* **10**, 278-291 (1966).
34. H. Y. Chang, E. Manning, and G. Metze, *Fault Diagnosis of Digital Systems*, Wiley-Interscience Publishers, New York, 1970.
35. F. F. Sellers, M. Y. Hsiao, and L. W. Bearnson, "Analyzing Errors with the Boolean Difference," *IEEE Trans. Computers* **C-17**, 676 (1968).
36. D. C. Hitt and R. J. Woessner, "Universal System Service Adapter," U.S. Patent 3,585,599, June 15, 1971.
37. E. B. Eichelberger and T. W. Williams, "A Logic Design Structure for LSI Testability," *Proceedings Workshop on Design Automation*, New Orleans, 1977, p. 462.
38. M. Y. Hsiao, "Hardware Error Detection and Failure Isolation Design Evaluation Technique," Invited talk, Fault Tolerant Computing Symposium-9, June 20-22, 1979, Madison, WI, and IFIPS-TC, September 1979, London, England.
39. J. P. Harris, W. B. Phillips, J. F. Wells, and W. D. Winger, "Innovations in the Design of Magnetic Tape Subsystems," *IBM J. Res. Develop.* **25**, 691-699 (1981, this issue).
40. D. T. Brown and F. F. Sellers, "Error Detection and Correction Features," U.S. Patents 3,508,194, 3,508,195, and 3,508,196, April 21, 1970.
41. D. T. Brown and F. F. Sellers, Jr., "Error Correction for IBM 800 bit-per-inch Magnetic Tape," *IBM J. Res. Develop.* **14**, 384-389 (1970).
42. M. Y. Hsiao and J. T. Tou, "Application of Error Correcting Codes in Computer Reliability Studies," *IEEE Trans. Reliability* **R-18**, 108-118 (1969).
43. P. A. Franaszek, "Sequence-state Methods for Run-length-limited Coding," *IBM J. Res. Develop.* **14**, 376-383 (1970).
44. A. M. Patel and S. J. Hong, "Plural Channel Error Correcting Apparatus and Methods," reissue patent, U.S. Patent Re 30,187, January 8, 1980.
45. A. M. Patel and S. J. Hong, "Optimal Rectangular Code for High Density Magnetic Tapes," *IBM J. Res. Develop.* **18**, 579-588 (1974).
46. H. C. Hinz, Jr., "Enhanced Error Detection & Correction for Data Systems," U.S. Patent 3,639,900, February 1, 1972.
47. E. G. McDonald and A. M. Patel, "Error Detection Systems," U.S. Patent 3,786,439, January 19, 1974.
48. Arvind M. Patel, "Error Recovery Scheme for the IBM 3850 Mass Storage System," *IBM J. Res. Develop.* **24**, 32-42 (1980).
49. J. M. Harker, D. W. Brede, R. E. Pattison, G. R. Santana, and L. G. Taft, "A Quarter Century of Disk File Innovation," *IBM J. Res. Develop.* **25**, 677-689 (1981, this issue).
50. W. W. Peterson, *Error Correction Codes*, MIT Press, Cambridge, MA, 1961, Ch. 10.
51. D. C. Bossen, "b-Adjacent Error Correction," *IBM J. Res. Develop.* **14**, 402-408 (1970).
52. M. Y. Hsiao and K. Y. Sih, "Serial-to-Parallel Transformation of Linear Feedback Shift Register Circuits," *IEEE Trans. Electron. Computers* **EC-13**, 738-740 (1964).

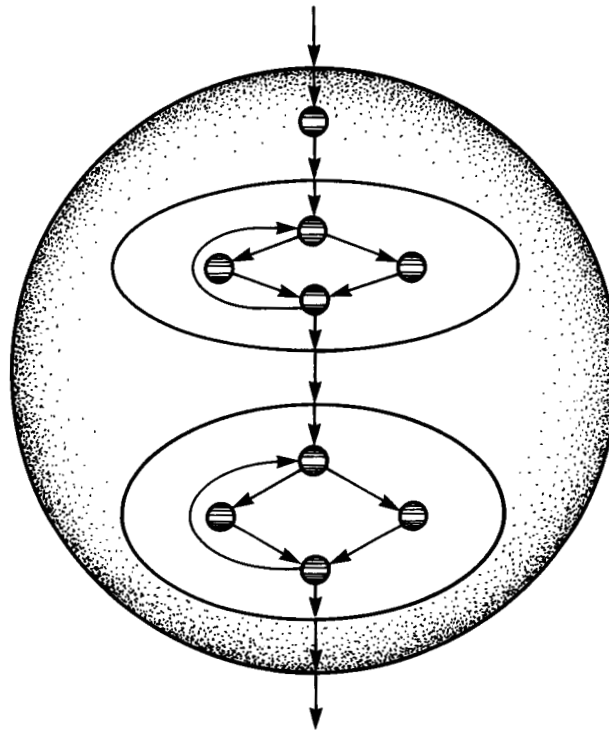
Received May 2, 1980; revised August 25, 1980

M. Y. Hsiao is located at the IBM Data Systems Division laboratory, Poughkeepsie, New York 12602. W. C. Carter is located at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York 10598. J. W. Thomas is with the Data Processing Products Group at the IBM laboratory in Poughkeepsie, New York 12602. W. R. Stringfellow is located at the IBM Field Engineering Center, Research Triangle Park, North Carolina 27709.





Software Technology



Directed flow graph

The *Journal* acknowledges the contributions of S. P. Carter to the acquisition, review, and editing of the papers in this section.

Editor