A study of intermittent faults in
digital computers

by ÖMÜR TAŞAR and VEHBİ TAŞAR

University of Detroit
Detroit, Michigan

ABSTRACT

Definition of intermittent faults in digital computer systems and their possible causes are given. The effects of intermittent faults on the performance of systems and alternatives to overcome these effects have been examined. Present attempts to cope with the intermittent failures and future research areas have been explored.

INTRODUCTION

The advanced system architecture of the current computers has had an impact on reliability and maintainability concepts. Although there have been evolutionary inventions in hardware and software built into computers, the complexity of the whole system makes the problem of fault isolation difficult. It is of vital importance to keep a system running as well as producing the system with the utilization of the best engineering knowledge. Hence one should be concerned with providing high availability and easy maintainability at the design stage. Due to insufficient attention devoted to this aspect, many of the machines in the field today have been suffering from poor reliability and maintainability.

In the so-called space age, it is hard to believe that systems can go down unexpectedly or cause interrupts for no apparent reason and groups of people wander around for many hours, even days to find the problem. All the maintenance procedures embodied in the machine do not give any hint. The field engineer does his best and the machine does not respond. At some point in time, the machine chooses to run due to an unknown action and there is almost no information on why, how, where and when this happened. This dramatized description is quite realistic for a number of current installations.

The problem causing the above situation is called the intermittent fault. In most systems 80 to 90 percent of faults are estimated to be intermittent. These faults account for more than 90 percent of total maintenance expense because they are difficult to detect and isolate.

INTERMITTENT PROBLEM DEFINITION

Intermittents were first defined as faults over which the user had little or no control. This definition still holds true. Today some people define intermittents as failures that are not reproducible since the conditions causing or surrounding an intermittent are often not known. The inherent inconsistency in the occurrence of such faults is a major hindrance to a detailed analysis of these conditions. Intermittents can be defined as random failures that prevent the proper operation of a unit for a short period implying that the duration of failures is not long enough for the application of a test procedure designed for permanent faults.

Some intermittents are known to occur due to external effects such as temperature, humidity, vibration, power fluctuation, pollution, pressure, and electromagnetic fields. Assuming that the system runs under specified external conditions, this class of faults constitutes a small portion of intermittents.

The nonenvironmental intermittents, namely the faults that are within the system, seem to be the basic source of irritation to the user. There is a difference of opinion as to the behavior of this class of intermittents. Some believe there actually exists a permanent failure which is stimulated only under a certain sequence of events; consequently, it appears as intermittent. The other opinion is that portions or components of the system malfunction intermittently. For example, loose connections, resistance variations and partially defective components may cause such faults. Examples of both opinions have been observed in practice. There are also intermittents that eventually become permanent failures. Such intermittents are caused by deteriorating or aging components. Once they become permanent, they can be detected by existing procedures. However this transition can take from a few minutes to several months during which the frequency of intermittents increases intolerably. To speed up this transition, environmental conditions can be drastically changed to unfavorable levels. Sometimes this technique, known as the stress technique, works but it may inflict new damage in other parts of the system.

The source of nonenvironmental intermittents can be
software as well as hardware. It has not yet been established which causes the major part of such intermittents. However, the general trend is to treat them as hardware oriented.

REVIEW OF LITERATURE ON INTERMITTENT FAULTS

The first experiments on the effects of intermittent faults were conducted by Ball, Hardie and Suhocki in 1966. A highly sophisticated logic simulator was developed for the purpose of normal and/or fault simulation of the Saturn V Launch Vehicle aerospace computer which was being designed by IBM. The computer is a binary, fixed point, serial machine employing triple modular redundancy to provide very high reliability. The simulator was capable of analyzing single and multiple, permanent and intermittent faults. Nearly 800,000 intermittents were simulated to obtain a reliable statistical sample. The duration of these intermittents varied between 500 nanoseconds (one clock time) to five milliseconds. They were injected at randomly selected points of combinational and sequential logic circuits in the arithmetic-instruction and multiply and divide units, at the execution time of the simulator.

The analysis of the intermittents included a record of the time of occurrence, time and number of detections and the number of intermittents that caused a difference from the correct output. The probability of detection was calculated based upon this information.

The authors revealed interesting as well as important conclusions from the stimulation output. These can be summarized as follows. Only 8.3 percent of the total intermittents injected caused the computer to perform incorrectly. The combinational logic was less sensitive to intermittents. In other words, the probability of detection in combinational logic was smaller compared to sequential logic. A single intermittent fault of one clock period was almost undetectable. Single intermittents of longer duration had very low probabilities of preventing the correct operation. The probability of detection was directly proportional to the duration of intermittents and highly dependent on computer modules.

The authors experienced that field failures in aerospace computers tend to be intermittent. In commercial computers, intermittents have been estimated to constitute 90 percent of all field failures. Hence, maintenance cost is mainly due to intermittents. The authors believed that most intermittents were due to insufficient testing at the factory. They proposed to develop better detection procedures as a partial solution to the intermittent problem. The other approach was to design equipment insensitive to intermittents introducing retry and/or redundancy.

IBM has also tried to deal with intermittents in systems 360 and 370. Here the objective was to reduce the number and length of unscheduled maintenances and to minimize the impact of intermittents on system availability. Automatic error recording at the instant of discovery followed by error recovery was proposed as a feasible solution. The so called Recovery Management of IBM follows several steps to achieve the above objectives.

Instruction retry can be affected in the I/O area, central processor and main storage areas. Selective termination enables the system to examine the failing environment while all other jobs are running. These functions are incorporated in Recovery Management in four steps. First, a functional recovery is attempted. If the retry of the interrupted operation is successful, the fault becomes transparent to the user. If not, the next function is System Recovery where a selective termination is effected to analyze the failure. Then the system-supported restart is tried without stopping for repair. If a stop is required, system repair utilizes all the detailed error analysis records. To perform all these steps, Recovery Management has I/O Device/Unit Recovery, Channel Recovery, I/O Recovery Management, CPU/Processor Storage Recovery, System Associated Recovery and Error Record Retrieval facilities.

Honeywell's proposal to overcome the intermittent problem was also retry. The Honeywell 6000 was implemented as a retryable processor. Maestri discusses the problems encountered and the design proposal to avoid such problems. Some instructions cause a destructive read of a memory location. Therefore, it is necessary to restore that memory location before retry can be attempted. For this purpose, a buffer register has to be added. Data can be held in this buffer register until error detection and correction codes report correct data recovery. Instruction retry for a MOVE may cause problems if a data block overlays another block, partly destroying the latter. To avoid this, snapshot registers are added where at the occurrence of a fault the state of the cycle control flags and address register could be saved. Snapshot registers can also be used as a diagnostic aid. To retry instructions with indirect addressing, several methods are proposed. One way is to obtain a pointer to the first indirect word. Restoring should take precautions against parity errors in the updated word and double updating. The second approach requires a scratchpad memory where the state of sequence control flags and memory addresses are saved for every cycle of an instruction. Software can then handle restoration of indirect words assuring no error. Multicycle instructions change the contents of registers with the speed of the adder cycle. In order to retry such instructions, intermediate registers are needed to protect the contents of primary registers and to keep up processor speed. These can be placed at the inputs of the adder on lines from memory and primary registers. Data can be held here until error checking is completed.

To solve problems due to instruction overlap, four instruction counters are added for simultaneously executed instructions. If an error occurs, the instruction counter of the failing cycle can be identified either selectively or by a program utilizing a failure flag included in the scratchpad memory.

The addition of the above mentioned registers increases the cost of implementation. However, the new processor is almost 100 percent effective in doing instruction retry. The cost advantage can be justified bearing in mind that instruction retry is 80 to 90 percent successful. Honeywell now...
A Study of Intermittent Faults in Digital Computers

Some extract of the text follows:

even states, in sales literature, that the 6000 system is capable of handling 95 to 97 percent of all intermittent failures.

The other approach to coping with faults in digital computers has been the introduction of redundancy. Research has shown that Triple Modular Redundancy (TMR) can mask solid as well as intermittent faults. If TMR is applied at the module level, then the effect of a single intermittent can be permanent in a sequential circuit. In case an intermittent induces a faulty state, it must be corrected with a resynchronization sequence before a second failure occurs. TMR has been used in some space projects; however, there has been no commercial application.

Totally self-checking check circuits have been designed. Such circuits are capable of detecting failures in themselves during normal operation. If intermittents last long enough, they affect the output in such a way that they can be detected.

Theoretical modeling of intermittent faults has been attempted in recent years. Breuer presented a two-state first order Markov model to represent the intermittents. Figure 1 shows the two states: the faulty state (N1) and the normal state (N2).

Since the Markov model is probabilistic, each state and every transition is associated with probabilities. With respect to intermittents, a circuit either operates normally or it possesses an intermittent. Let p be the probability of having an intermittent, then the normal state will have a probability of (1-p). To apply this model to any circuit, first of all the parameters, namely p, r, s, have to be estimated.

A time interval (T, T') is considered for fault analysis. Fault patterns are defined over this interval as follows: d=(d0, d0-1, ..., d0, d1). The subscripts refer to the number of clocks in this interval. d=(0, 0, ..., 0, 0) implies a normal circuit. d=(1, 1, ..., 1, 1) represents a solid fault. d=(0, 0, ..., 1, 0, 0) is the fault pattern of an intermittent occurring in the third clock period. The probability of each fault pattern can be calculated using the Markov model. A complete test set will be obtained if test sets are generated for all possible fault patterns. A modified algorithm is proposed for test generation that makes use of two types of gates called d-OR and d-AND. d-OR has the property that its output is a d, if and only if at least one of its inputs is a d or d; otherwise its output is 0. In other words, if any input of d-OR receives a fault, it can propagate it. The output of d-AND is a d, if and only if all of its inputs have a d; otherwise, it is a 0. Test generation requires that the combinational logic be evaluated the same number of times as the number of fault patterns present in a sequential circuit. In sequential logic analysis, the circuit evaluation should be repeated for every fault pattern in every clock period within the time interval (T, T'). Hence, even for a small circuit the test generation is a very lengthy procedure.

Every test generated has a certain probability of detection. If a confidence level can be specified, then a lower bound for the number of tests that are required to detect an intermittent can be calculated.

Kamal's model is also probabilistic. It is based on pattern recognition techniques. The analysis covers well behaved signal independent single intermittents in non-redundant circuits. A state space includes the state of being faultfree—the states possessing an intermittent that do not affect the output and the states possessing an intermittent that affect the output. The test set is the set of tests that are generated to detect permanent faults. The proposed solution is the repeated application of these tests. It is proven in the paper that an intermittent can be detected by infinite number of repetitions of a test. The probabilistic model is utilized to find the finite number of repetitions satisfying a confidence level. The model requires estimation of the probability that an intermittent occurs and the conditional probability of the output being affected given an intermittent has already occurred. The probability of detection given an intermittent is calculated according to Bayes' rule. After each application of a test, this figure is updated. This probability approaches 1 with an infinite number of applications. However, the procedure can be stopped once the confidence level is reached or the number of repetitions can be calculated in terms of the confidence level and the other probabilistic parameters. In most cases this number is very large. Kamal provides an optimization procedure by the use of integer programming.

Diagnosis procedure follows the same approach. Here it is assumed that an intermittent has been detected. Using the test sets and the set of permanent faults, a coverage table is generated. From this table a test set is chosen that covers the permanent faults. These tests are repeatedly applied until one fails. This reduces the fault table and the possible faults. The above cycle is repeated until the fault set consists of a single fault. If there exists no test set to cover all possible faults, the diagnosis experiment is again terminated. The length of the experiment is calculated making use of the model and is shown to be finite. The fault table is also analyzed to obtain minimum diagnostic resolution. Optimization of the diagnosis procedure is not offered.

There are two major difficulties in the application of Breuer's and Kamal's proposals. First, it is very hard to obtain realistic estimates of the model parameters using the current available data. Even if an extensive experiment is conducted on a certain system, the parameters would apply to this particular system. The unpredictable behavior of intermittents may also impose an update on the parameters. The second problem is the huge number of tests required to detect and diagnose intermittents. In this respect, the proposed procedures may not be economically feasible. Even the detection of solid faults are far more expensive then desired.
Intermittent faults have been analyzed at the circuit level. Yen discusses intermittents due to noise in four phase MOS gates which are widely used in LSI circuits. The problem is a charge redistribution problem. It is possible that the charge at some output node will decrease to critical levels causing intermittents. The situation does not fit the stuck-at model and testing will not be successful in detecting this type of failure. Yen proposes the addition of prechargers to avoid the problem. The number of the prechargers can be optimized by the use of computer-aided circuit simulation in order to decrease the additional hardware cost.

ALTERNATIVES FOR SOLUTION

No ideal method has been established to diagnose intermittent faults so far. Academic work offers solutions on restricted models that do not realistically represent the intermittent space. Industrial efforts concentrate on increasing the availability, hence they are mainly concerned with the minimization of the effects of intermittents. Up to now, the former has found no application. On the other hand, commercial manufacturers introduced retryable systems that recover from intermittents in a very short period. Although retryability is not a method to diagnose or to avoid the intermittents, it seems to be the only feasible way of recovering from such faults with minimum impact. Indeed, it is difficult to detect and diagnose something that goes on and off unpredictably, and retryability seems to be the only solution at the systems level currently and in the near future. Interruptions at the circuit or card level should be viewed as a completely different problem. At factory testing, intermittents account for only 30 percent of all faults, whereas in the field they are estimated to constitute 90 percent of all faults. Furthermore, the consequences of intermittents occurring at the system level are far more serious and costly compared to their effects at the testing level. Therefore, our efforts should concentrate on methods dealing with system intermittents. Here we discuss retry and other possible alternatives and where and how they can be effective in fighting intermittents.

Instruction retry is not a new concept. It has been a regular procedure that is carried out whenever an error occurs in the I/O area in reading or writing a tape. The new attempt is to extend this feature to central processor and memory units. IBM has been working on this since 1967; Honeywell since the early 70's. Burroughs has recently expended efforts on making its large scale processors retryable. The outcome indicates that these systems can survive 95 percent of intermittents. 100 percent effectiveness is also possible. Since the chance of success in retrying is .8 to .9, the system will have 75 to 85 percent less faults that are apparent to the outside world. In other words, instead of giving an interrupt, the system continues to operate successfully with a few microseconds or milliseconds of delay, caused by the execution of retry. Recovery alone does not help to isolate the intermittent, but an immediate remedy is achieved with respect to the operation of the system. Adequate error recording serves as a diagnostic aid during scheduled maintenance. Utilization of these records toward a test program will be discussed later.

Retry requires both hardware and software support. Intermediate and snapshot registers and scratchpad memories have to be added in order to be able to save the status of the system prior to an intermittent. Only then can the system go back and retry the failing state. Even with the addition of the sophisticated retry software, the implementation is not costly compared to the maintenance cost due to intermittents.

Today it is common practice to use stress testing to change an intermittent to a solid failure. Marginal voltage and timing conditions can be set or thermal stress can be applied such that slow rise times, low switching thresholds and race conditions are amplified. If test and diagnostic programs are run during the application of mechanical stress, faults due to loose connections, defective connectors or printed circuits can be caught. The disadvantage of this method is the dedication of the processor to maintenance procedures as well as the possible infliction of new damage due to thermal and mechanical stress. By all means retry is a better approach than this technique.

Retry is aimed at continuing the operation of the system. Whether it is successful or not, a detailed error recording is a major part of this scheme. If retry is not successful, the analysis of the error recording prior and at the instant of error occurrence would assist the personnel to determine the cause of the error in an easier and faster way. If retry is successful, then the error recording dump would be analyzed during scheduled maintenance to determine the cause of the intermittent. It may be advantageous to replace or repair the part causing the intermittent to avoid future possible intermittents. Although retry requires a short amount of time, it is more desirable not to have any interruptions. If this policy is continued persistently, the rate of interruptions due to intermittents may gradually decrease.

Another important outcome of detailed error recording is the possibility of a test program that diagnoses intermittent faults. The accumulated error recording of a certain period may well include information about the most frequent intermittents such as the rate of occurrence, the effect of the intermittent, the symptoms and the fastest correction procedure. Similar to the detection procedure, given an intermittent and its symptom(s), we can run the above mentioned program and come out with an intermittent or a set of possible intermittents that can be examined. Isolation of the intermittent will be an easy matter if it actually belongs to the group of the intermittents under consideration. If not, the intermittent can be located by existing means and the test program can be expanded to include the new situation. As time progresses, one may come up with a complete intermittent diagnostic routine.

Realization of a diagnostic routine for intermittents requires extensive error recording on a retryable processor. Diagnosis can be achieved by another practical method called dynamic monitoring. As the name implies, dynamic monitoring involves a continuous scan of the machine to be able to detect a fault at the instant of occurrence. Other-
wise, there is practically no way of reproducing the same event for purposes of detection and diagnosis. This scheme can be realized by placing test points and interrogating the values of these test points at every clock period. If the machine has retry capability, dynamic monitoring can be very successful in isolating intermittents. The values of test points can be continuously stored in a monitor memory, the size of which depends on the number of test points and the number of clock periods that have to be considered. In case of an interrupt, one of the test points will have a faulty value. These values can be stored in a memory for further reference. If the retry is successful, the particular test point will have the correct value. Comparing the two maps of test points before and after retry, the faulty test point can be isolated. To locate the fault causing component, it would suffice to examine the components feeding that test point.

Although dynamic monitoring requires test points and a memory of considerable size, the advantages are apparent. The fault is actually isolated during the operation of the system. All intermittents are detected regardless of their cause. If statistics are maintained, deteriorating components can be isolated and future faults predicted.

The feasibility, development and application of this technique has to be considered in conjunction with retry capability. If retry is not available, the technique can be implemented by straight dumping of the recorded memory. The basic problem in realizing this scheme is the optimal location of the test points.

The above discussion refers to the solution of intermittents at the systems level. Dynamic monitoring can also be employed in testers that are used to screen circuits and cards. At the circuit level, the models presented by Kamal and Breuer may also be applicable. Current automatic testers operate at a very high speed. Hence, a very large number of tests can be applied in a short period. If careful factory screening is desired, these techniques can be incorporated with the testers. Although the cost of testing will increase, less faults will escape the factory screening. Economic feasibility of these two models can be studied by the use of computer-aided simulation.

ACKNOWLEDGMENT

The authors wish to thank Mr. R. E. Stackhouse for his guidance and comments during the preparation of this paper. This work was supported by the Burroughs Co. through an internship program.

REFERENCES
