amdahl

"A TECHNIQUE FOR FAILURE ISOLATION IN A LARGE COMPUTER SYSTEM"

A. Shah

Amdahl Corporation

TECHNICAL PAPER

"A TECHNIQUE FOR FAILURE ISOLATION IN A LARGE COMPUTER SYSTEM"

A. Shah

Amdahl Corporation

Presented at:

IEEE International Conference on Circuits & Computers (ICCC)

October 1 — 3, 1980

Port Chester, New York

"A TECHNIQUE FOR FAILURE ISOLATION IN A LARGE COMPUTER SYSTEM"

A. Shah

Amdahl Corporation

Sunnyvale, California

ABSTRACT

This paper describes a technique implemented on the Amdahl 470 computers for failure isolation. The 470 system includes a number of hardware elements which facilitate fault diagnosis and isolation of the failing part. The algorithms as well as the hardware and software components of the technique are discussed.

INTRODUCTION

Fault diagnosis and repair of a large computer system is a difficult problem. This problem arises both during the manufacturing process and in the field. In both cases, it is necessary to isolate the failure and carry out the repair as quickly as possible. The repair consists of replacing a unit containing the failing part. During manufacturing, the replaceable unit is usually a chip, whereas in the field, it is usually a card (especially for a large mainframe). Thus the failure has to be isolated to a replaceable unit. For a large number of complex, highly interconnected parts, this is a time consuming and expensive process. The failure isolation is typically achieved using a combination of hardware and software tools. The hardware tools consist of error checking circuits, which can indicate the approximate region of failure; as well as other debugging aids which can be used manually or by software programs for isolation. The software tools consist of diagnostic programs ranging all the way from large system exercisers and error analyzers to microdiagnostics and fault locating tests [3]. In most cases however, failure isolation requires a large degree of skill and luck.

This paper describes a technique that has been implemented on Amdahl 470 computers for solving this problem. The hardware and software required to implement it are also described.

AN OVERVIEW OF THE TECHNIQUE

The method presented here relies on the fact that fault detection is much easier than isolation. Most large scale computer systems use error checking circuits which can detect errors in the data and control paths. In addition, diagnostic programs designed to detect errors are much simpler than those designed to isolate faults. In fact, much of the testing can be carried out at the architectural level, that is, using only machine instructions (as opposed to microdiagnostics, etc.) so that the same diagnostics can be used across a range of models with the same architecture.

Now, given that most errors can be detected by the hardware and diagnostic programs, the problem becomes one of fault isolation. The error is detected and reported some number of machine cycles after the failure manifests itself. In addition, the detection could be in a component far removed from the failure. Thus it is necessary to narrow the region of error detection in time and space.

This method accomplishes that objective by determining the first machine cycle where the failure manifests itself as an error in the machine state, and by determining the portions of the machine state in error.

The method consists of three steps. In the first step, various diagnostic tests are executed on the failing machine until the error is detected by a diagnostic. In the second step, the same diagnostic test is executed cycle by cycle in a controlled, deterministic environment. On each cycle, the state of the machine is compared with a predetermined expected state. In this way, the region of error detection is narrowed down (in time) to one machine cycle. In the third step, the region is narrowed down in space by identifying the miscomparing substate.

In effect, this method compares a good machine with a failing machine. This idea is not new; in fact, von Neumann suggested it in 1946 in one of his original papers [1]. Other methods in use include applying test patterns and comparing the machine behavior to the behavior predicted by simulation [3]. The method presented here avoids the time consuming test pattern generation and simulation for the complete system. Instead, it makes use of the error detection capability of diagnostic programs which are provided in any case for testing and verification. There are many unique features of the hardware and software design which make possible the implementation of this technique. The rest of the paper describes this implementation.

HARDWARE

The 470 hardware includes a number of features designed to facilitate diagnosis and maintenance of a machine. This section will discuss the main features used for failure isolation.

1. Console

The 470 console is a minicomputer system which is capable of performing a number of hardware control functions in addition to the normal console functions. A block diagram of the console system and its interface to the 470 is shown in Figure 1. As shown in the figure, the console system consists of a CPU and memory, two floppy drives, a fixed head disk, a CRT/keyboard, and a modem link. The floppy drives are used for loading diagnostic programs. The fixed head disk contains the console operating software and the software to implement the algorithms described in the next section. The CRT/keyboard provide the user

interface for performing console functions as well as for loading and executing the diagnostics.
The modem link is used to achieve remote monitoring and control of the 470. With the link, a person at a central site can perform all the functions that can be performed locally at the keyboard. In addition, this link can be used for data transfer. As we shall see in a later section, this is quite important for the isolation technique.

The tape drive shown is not a normal part of the system. It is attached to the console when needed for data collection.

Finally, the console interfaces to the 470 via the Console to Computer Interface Processor (CCIP). This interface allows the console to have complete control over the operation of the 470. The functions that can be performed across this interface will be described later.

2. Scan Out

For fault diagnosis, one of the most important elements is the ability to observe the internal states of the machine. The 470 provides the ability to read out the values of most of the latches in the system, this process being known as scan out. In addition to being useful for manual trouble shooting, this feature is central to the failure isolation technique since the state of the machine at any given time is determined by "scanning out". As described later, it is possible to select any set of latches and scan them out via the CCIP. The scan out is completely benign in that is does not alter the state of the machine.

3. Single Cycle

For determination of the first clock cycle where expected and actual machine states differ, the diagnostic program is executed cycle by cycle with a scan out following each cycle. It is necessary that the operation of the program in this mode be identical to the operation when executed at speed, so that the error occur again. There are some classes of failures which only cause errors when operating at speed (e.g, race conditions); however, the number of such failures is reduced greatly if the operation in both cases is identical. Note that operation is used here in a much stronger sense than just the program functional behavior; the machine has to sequence through identical states in both cases. The 470 is designed to achieve identical operation at speed and in single cycle mode. The console, via the CCIP, can control the clocks and issue single clock pulses.

4. CCIP

The console to computer interface, and the associated processor provide the console with the ability to control the 470 with functions such as start, stop, reset, and single cycle. The console can also load diagnostic programs into the 470 memory via this interface.

In addition to the above, the console can scan out any selected set of latches using the CCIP and the "mask memory". This memory is as large as the scan address space of the machine. By setting appropriate bits in the memory, the console can define which latches are to be scanned out. The scan out is done by the CCIP when the appropriate command is issued by the console. This allows a high speed scan out operation.

The CCIP is also capable of generating a "checksum" of the latch values scanned out. The checksum is a compact representation of the machine state, although of course information is lost as a result of compaction. The checksum generation algorithm is chosen such that most failures which cause a deviation in the machine state will also cause a deviation in the checksum.

The next section describes how all these hardware elements are used to implement the technique.

ALGORITHM DESCRIPTION

The algorithm consists of two parts: data collection, and failure isolation. Data collection is performed on a known good machine, whereas failure isolation is performed on a failing machine. In both cases, the algorithm goes through identical steps as far as the 470 is concerned.

1. Data Collection

Data is collected for each diagnostic program. Before the program is executed, it is necessary to put the machine in a known, deterministic state so that valid comparison can be made during isolation. A program known as deterministic reset [2] is executed to achieve this known state.

After the reset, a scan mask is loaded into the CCIP memory. This mask selects the latches in the region which the diagnostic is testing. Then the diagnostic is executed cycle by cycle through the machine instructions where the testing is being performed. After each cycle, the selected machine state is scanned out. At the same time, the checksum is generated. The scan data and the checksum are written to the tape unit. An off-line process extracts the checksums from the tape and puts them on the floppy containing the diagnostic. Space limitations make it impossible to keep the complete scan data on the floppy disk. However, as we shall see in the next step, the checksum is adequate to carry out the isolation.

2. Failure Isolation

As mentioned before, diagnostic programs are executed on the failing machine until one of them detects an error. At this stage, we are ready to apply the second part of the algorithm. Deterministic reset is executed, the mask memory is loaded, and then the diagnostic is executed in exactly the same fashion as during data collection. This ensures that the machine will sequence through the same states again until the failure manifests itself as an error in the value of a latch. After each cycle, the checksum is generated and compared with the expected value obtained from the floppy disk. This is continued until a miscomparison occurs, at which point we have succeeded in narrowing the fault window to one cycle. This in itself is quite valuable, since manual isolation of the replaceable unit is now considerably easier.

The next step, if necessary, is to determine the latches in error. The expected scan values of the latches are not present on the console; this is where the remote link comes into play. The complete machine is now scanned out and the data is transmitted to a central site where the collected data tapes are available. In addition, the central site has complete description of each latch in terms of its function and interconnections. The scan data is compared with the expected values and the miscomparing latches are determined. The latch descriptions are transmitted back to the failing machine site where they can be used for isolation. It is now much simpler to isolate the replaceable unit, knowing the cycle where the error occurs, and the latches in error.

SUMMARY

This paper has described a failure isolation technique for use on a large and complex computer system. It solves this difficult problem by making use of the hardware features built into the machine and the error detection abilities of diagnostic programs. It avoids the need to produce fault isolation tests which require an expensive software effort.

As circuit densities increase and computer systems become even more complex, the problem of failure isolation will get worse. It will also be necessary to reduce the mean time to repair since servicing costs will otherwise increase drastically. It will be imperative that the hardware include more elements designed to improve diagnosability, and that techniques such as this one be used to take some of the black magic out of the process of repairing computers.

ACKNOWLEDGEMENTS

A large number of people, both in software and hardware, participated in the conception, design, and implementation of this project. I would like to thank them all for their contributions and for the enormous amount of enthusiasm, dedication, and creativity which they brought to the task.

REFERENCES

(1) A. W. Burks, H. H. Goldstein, and J. von Neumann, "Preliminary Discussion of the Logical Design of an Electronic Computing Instrument," report prepared for U.S. Army Ordnance Dept., 1946, in A. H. Taub (ed.), Collected Works of John von Neumann, vol. 5, pp. 34-79, The Macmillan Company, New York, 1963.

(2) M. Adham, T. J. Kenville, and G. L. Watson, "Deterministic Reset for the Amdahl 470 V/6 Computer," Proc. of the 1980 International Symposium of Fault Tolerant Computing (to be published).

(3) H. Ito, I. Ohata, A. Nakamura, and J. Nose, "Some Experiences in FLT System Implementation," Third USA-JAPAN Computer Conference, 1978, pp. 258-262.

(4) M. A. Breuer, and A. D. Friedman, Diagnosis and Reliable Design of Digital Systems, Computer Science Press, 1976.

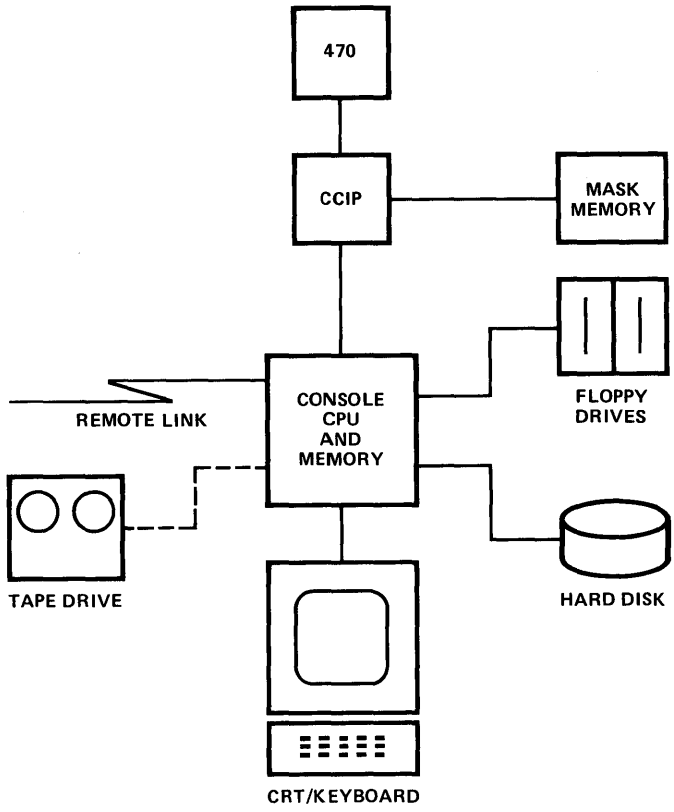

FIGURE 1