amdahl

"DETERMINISTIC RESET FOR THE AMDAHL 470 V/6 COMPUTER"

Mughith Adham, Thomas J. Kenville
and Gentry L. Watson

Amdahl Corporation

TECHNICAL PAPER

"DETERMINISTIC RESET FOR THE AMDAHL 470 V/6 COMPUTER"

Mughith Adham, Thomas J. Kenville
and Gentry L. Watson

Amdahl Corporation

Presented at:

Fault Tolerant Computing Symposium — 10 (FTCS-1-)

October 1 — 3, 1980

Kyoto, Japan

"DETERMINISTIC RESET FOR THE AMDAHL 470 V/6 COMPUTER"

Mughith Adham, Thomas J. Kenville, and Gentry L. Watson

Amdahl Corporation

The Deterministic Reset Test is a diagnostic test program that was developed for the AMDAHL 470 V/6 computer. This paper describes the development and usage of this test.

The test executes in the diagnostic processor used to monitor and control the state of the 470 V/6 mainframe. The objective of the test is to identify any hardware problems that prevent the initialization of the mainframe to a known state. This known state is used as a basis for fault isolation in subsequent diagnostic programs. The sequence used by the test can reset the machine from any initial hardware state and has embedded in it some 4000 different scan-outs of latch values. The sequence ends with a final scan of approximately 8000 latches. At any point during the test sequence, only the latches that have been initialized at that point are scanned out. The determination of the latches initialized at each cycle was performed by off-line computer analysis of scan data.

INTRODUCTION

Part of the diagnostic strategy for the AMDAHL 470 V/6 computer was to develop tests capable of detecting any deviation of the internal machine state from its expected behavior. To accomplish this, the hardware response to a test must be predictable and repeatable. The above requirement can be assured if the test always starts from the same initial machine state because the 470 V/6 executes synchronously. This initial machine state is achieved by the use of the 470 V/6 console processor in conjunction with special console-to-mainframe interface hardware.

The deterministic reset sequence is designed to initialize the machine to a state in which most of the mainframe latches and memories have known value. This sequence is used by other diagnostic tests for initializing the machine hardware. The Deterministic Reset Test hereafter referred to as "the test," identifies any hardware failure which prevents the reset sequence from executing successfully. It observes and analyzes the machine latches after each step of the sequence. The test can be thought of as a 4000 line logic analyzer monitoring and comparing the hardware during the initialization operation. It executes in the same fashion regardless of the initial machine state. This allows restarting the test after it detects a failure and stopping it one or more cycles prior to the failure in order to identify its source.

The test uses a bootstrapping technique, meaning that it uses the hardware initialized in earlier steps to initialize additional hardware in subsequent steps. At any step of the bootstrap, only those latches having known values are observed for fault indication. All the to-be-scanned latch values are combined into a 16-bit checksum which is compared to an expected value stored in the test. The expected checksum values were obtained by running the test on a fault free computer. If the checksum miscompares, a remote database can be used to identify the names of the miscomparing latches. This database contains approximately ten megabytes and is stored on magnetic tape. The number of observed latches continues to increase as the test proceeds. To speed its overall operations, the test runs in two modes: detection and isolation. In the detection mode, one checksum is compared at the end of each function within the reset sequence. Once a miscompare is identified, the test is rerun in isolation mode and checksums are compared for each cycle within the failing function. This identifies the first miscomparing cycle of that function.

The comparison of actual to expect values in a diagnostic test is commonly employed by the industry. Some approaches [3] employ two identical units performing the same function, with a third unit continuously comparing them. Other approaches [4,5] employ hardware simulation and modeling to generate expected values.

The use of checksums for diagnostic purposes is discussed in [1,2]. The primary reason for its use is to reduce the size of the test's expected-values database.

HARDWARE ENVIRONMENT

The test executes in the diagnostic processor of the AMDAHL 470 V/6. This processor is a mini-computer supporting a fixed-head disk, two floppy drives, a modem, and a keyboard/display unit.

This processor interfaces to the mainframe via special-purpose hardware. This hardware allows diagnostic programs to scan-out the mainframe, alter its registers and memories, and control its operating state.

The 470 V/6 is built using Emitter Coupled Logic circuits. Most of the machine is packaged in Large Scale Integration chips mounted on Multiple Chip Carriers. External interface logic and some memories are built using Medium Scale Integration chips mounted on Basic Logic Cards. The main storage uses dynamic Metal Oxide Semiconductor technology.

The machine operates as a collection of independent functional units that interface using well-defined protocols. It can operate in the same fashion both when mainframe clocks are running or when clocks are stepped in single-cycle mode. The major units of the 470 V/6 mainframe are:

Instruction Unit: This unit has a 9-stage pipeline that controls the instruction execution flow of the machine.

Execution Unit: This unit has a 2-stage pipeline that performs the data computations of the machine.

Storage Unit: This unit has a 3-stage pipeline that supports the accesses to mainstorage for both real and virtual mode operations. It holds the High-Speed Cache.

Main Storage: The storage can be expanded to 16 Megabytes.

Channel Unit: This unit has a 16-stage circulating pipeline that controls the interfacing of peripherals.

Channel Diagnostic Processor: This is a micro-programmed diagnostic tool. It is permanently connected to the channel unit.

TOOLS

The following tools are a part of the overall diagnostic system of the 470 V/6 computer. They are used by the test as it is being run for fault isolation, and were also used during the test development.

Isolation Monitor: This is a system [6] running in the 470 V/6 diagnostic console. It interacts with the reset test to provide support for scan masks, checksums, and test positioning. The test passes scan mask specifications to the monitor which uses them to control the scanning operation. The test requests the monitor to perform a scan operation. The monitor, in turn, scans the machine and generates a checksum from that scan. The monitor then compares this value to an expected checksum for that cycle stored with the test. At the beginning of each step in the reset sequence, the test performs an identification call to the monitor. A monitor command can be used to position the test at any of these critical points.

Channel Diagnostic Processor: The Channel Diagnostic Processor is a diagnostic tool physically connected to the 470 V/6 channel as an external device. This processor is implemented using specially designed microcode. This device uses the 470 V/6 system clocks and therefore executes synchronously with the rest of the machine. The microcode is designed so that the device behaves identically whether the system clocks are running or stepped in single-cycle mode.

Console to Computer Interface Processor: This hardware allows the console to specify a sixteen-bit address and retrieve the latch value corresponding to that address. When checksums are being generated, values for all 64K scan addresses of the 470 are scanned out. A 64K masking memory specifies for each latch address if that latch value is to be included in the generated checksum. A sixteen-bit checksum is computed by exclusive or-ing all to-be-used latch values sixteen bits at a time. This ensures that any single-bit error is detectable.

SCOPE OF THE TEST

This section describes the different functions performed by the test. These can be divided into three groups: 1) reset and synchronization, 2) initialization of memories, 3) initialization of latches.

The reset and synchronization portion of the test issues a hardware reset to all the units and clears all the registers of the machine. This reset clears any error condition that may exist. It further initializes most of the control latches and synchronizes the various counters and sequential machines of the mainframe. This synchronization is essential if cycle-by-cycle operation is to be repeatable.

The initialization of the 470 V/6 memories is performed by console operations. The memories initialized are the main storage, the high-speed cache, the segment base register stack, the channel and subchannel buffer stores, the channel diagnostic processor microstore, and the translation lookaside buffer.

The latches to be initialized are mostly data path registers. To initialize these it is necessary to execute an operation that uses them since they are not cleared by the hardware reset. Other latches initialized by the deterministic reset sequence are latches that are dedicated for executing a given instruction (such as edit and mark). The only way of initializing these latches is to execute the corresponding instruction. The main operations performed for initialization latches are:

1. test I/O to the Channel Diagnostic Processor
2. start I/O to the Channel Diagnostic Processor to move data in and out
3. edit and mark
4. divide float and divide decimal
5. move character long
6. virtual mode operations

At the end of the execution of these operations, most of the CPU latches are at a known state. Figure 1 is a flow-chart of the reset test operations.

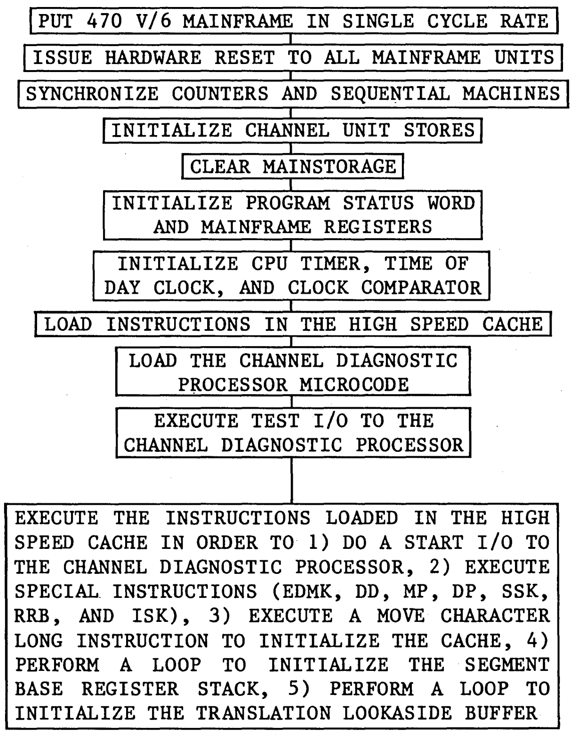
PUT 470 V/6 MAINFRAME IN SINGLE CYCLE RATE
ISSUE HARDWARE RESET TO ALL MAINFRAME UNITS
SYNCHRONIZE COUNTERS AND SEQUENTIAL MACHINES
INITIALIZE CHANNEL UNIT STORES
CLEAR MAINSTORAGE
INITIALIZE PROGRAM STATUS WORD AND MAINFRAME REGISTERS
INITIALIZE CPU TIMER, TIME OF DAY CLOCK, AND CLOCK COMPARATOR
LOAD INSTRUCTIONS IN THE HIGH SPEED CACHE
LOAD THE CHANNEL DIAGNOSTIC PROCESSOR MICROCODE
EXECUTE TEST I/O TO THE CHANNEL DIAGNOSTIC PROCESSOR
EXECUTE THE INSTRUCTIONS LOADED IN THE HIGH SPEED CACHE IN ORDER TO 1) DO A START I/O TO THE CHANNEL DIAGNOSTIC PROCESSOR, 2) EXECUTE SPECIAL INSTRUCTIONS (EDMK, DD, MP, DP, SSK, RRB, AND ISK), 3) EXECUTE A MOVE CHARACTER LONG INSTRUCTION TO INITIALIZE THE CACHE, 4) PERFORM A LOOP TO INITIALIZE THE SEGMENT BASE REGISTER STACK, 5) PERFORM A LOOP TO INITIALIZE THE TRANSLATION LOOKASIDE BUFFER

FIGURE 1. FLOW-CHART OF TEST OPERATIONS

The approximate number of cycles required for each function of the reset sequence are as follows:

1.	reset--------------------	1000
2.	mainstorage--------------	800000
3.	other memories-----------	70000
4.	registers----------------	800
5.	latches------------------	4000

Due to the large number of cycles involved only a subset are monitored. The selection of the cycles to be monitored is based upon the amount of new hardware involved in each new cycle. For example, only the loading of the first location in the cache is monitored, since loading of subsequent locations uses the same hardware. In general, most of the non-scanned cycles occur in the initialization of memories.

APPLICATION AND USE

Manufacturing System Test: After the individual components of the 470 V/6 are checked out, the computer is assembled and tested as a complete system. During such testing, more than one fault may exist in the machine. As the test executes many machine operations, it detects a substantial percentage of these faults (about 70 percent), and isolates them, one at a time. The test is particularly useful for isolating failures in the control logic part of the machine which are hard to isolate using manual techniques. In order to run the test, specially trained operators are needed and an on-line computer is required. These requirements are satisfied in the system test environment becuase the test and its support system is repeatedly used during the assembly of every machine.

Return Part Testing: Field maintenance activity results in the isolation of failures to the level of a field replaceable component. These are returned and individually tested to determine the exact cause of failure so that they may be repaired. Parts with faults that cannot be identified by individual component testing have to be tested on a working machine. The test is a powerful tool for this purpose because it can rapidly pin-point a failing component if the failure is within the scope of the reset sequence.

Field Test: Though intended for use at the customer's site, the operation of the reset test proved difficult in that environment. Many variations of machine design level, coupled with the test's extreme sensitivity to the internal operation of the machine logic, limited its use.

OPERATING PROCEDURES

This section describes the procedure used to detect and isolate a fault using the Deterministic Reset Test. The procedure has five major steps.

The first step is to execute a keyboard command that invokes the reset sequence from the console's fixed-head disk. This is a go/no-go operation taking about five seconds. The command performs a limited scanout of latches after the completion of its reset sequence. If the reset sequence does not initialize the machine, the test is loaded and run to isolate the cause of the failure.

The second step is to load the test floppy and run the test in fault detection mode. This step takes at most two minutes. Detection is accomplished by scanning after each function of the reset sequence until the failing function is identified. Once this identification is made, further isolation within that function can be attempted.

The third step involves rerunning the test in fault isolation mode. This step takes at most seven minutes. Isolation is accomplished in three steps: 1) the test is restarted, 2) all functions preceding the failing functions are executed, 3) the failing function is examined on a cycle-by-cycle basis.

This identifies the first miscomparing cycle in that function. To identify the miscomparing latches at that cycle, a connection must be made to the remote database.

The fourth step involves a connection to the Amdahl Diagnostic Assistance Center (AMDAC) over telephone lines. Once the datalink is established, an image of the machine state at the failing cycle is transmitted to an on-line computer system. This system compares the received image to an expected image and identifies the names of the miscomparing latches. These names point into the 470 V/6 logic diagrams so that the exact source of the failure can be traced. The time needed for this step is in excess of ten minutes. Figure 2 shows the connections between the different components used in isolating a machine fault.

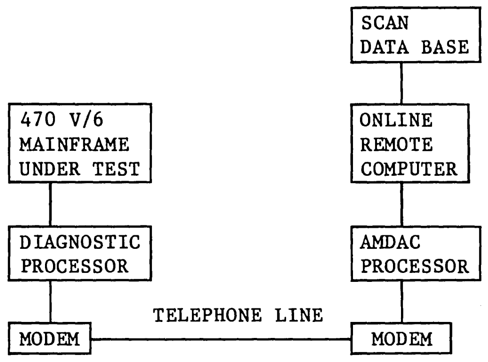

FIGURE 2. CONNECTIONS BETWEEN SYSTEM COMPONENTS

The fifth and final step involves a command that allows the user to re-execute the test and stop one or more cycles prior to the miscomparing cycle. At that time, various parts of the machine can be observed using the console formatted scan display in order to identify the failing component. The test may be run in a "forced isolation" mode on a fault free machine. In this mode, after each scan, the checksum is compared to its expected values. This takes approximately 10 minutes.

COVERAGE

The 470 V/6 contains approximately 17000 scannable latches. Of these, 8000 are initialized by the reset sequence. Latches not initialized by the sequence fall into three categories:

1. those that cannot be initialized without the use of external I/O devices, which number about 2000 latches,

2. those related to optional hardware features that do not exist on all tested 470 V/6s, which number about 10000 latches.

3. and those that differ in operation depending on the hardware design level of the 470 V/6 mainframe, which number about 6000 latches.

The final scan of the test includes all 8000 latches initialized by the reset sequence. The exact cycle of initialization is known for only 4000 because of limitations of the identification method used. Each one of these latches is scanned at the cycle it is first initialized and all cycles thereafter. The other 4000 latches are scanned only at the last cycle of the test.

TEST DEVELOPMENT

The first step of the development process was to study the machine design in order to identify the operations needed to initialize the latches of the machine. Design engineers were consulted to identify the operations required for each particular unit. The initialization of the channel unit required interfacing to an external device in a synchronous fashion. This led to execute synchronously with the mainframe. The result of this first step was the development of the deterministic reset sequence. This sequence was coded in a diagnostic test language and transferred to a floppy disk.

The second step of the development process was to identify which latches were to be scanned during the reset operation. This involved running the reset sequence from various initial machine states. These states were established by powering-down and powering up the machine without performing any power-on resets, and by running selected initialization routines and other system level programs. These functions were performed on many different machines. There were approximately 300 initial states used. For each of these "runs," the machine state was scanned-out at selected cycles. The scan information was recorded on magnetic tape and saved for later processing that would extract the first cycle at which each latch became deterministic. In this sense, a latch is said to have become "deterministic" at a given cycle of the reset sequence if for each subsequent cycle of the sequence the value assumed by the latch is consistent for all the runs.

The processing to identify the cycles at which latches became determined was based on three axioms:

1. A latch starting from any initial state is eventually forced into a deterministic state by the reset sequence, and

2. That latch is forced to a deterministic state at a unique cycle of the sequence, and

3. The state of every latch at the last cycle of the reset sequence is the same for all runs.

To find the cycle at which each latch became deterministic the analysis proceeded from the final cycle of the reset sequence towards the first, one cycle at a time. This was done in order to find the cycle at which the latch value ceased to be the same for all the runs. When the latch values in any two runs differed, the cycle _just prior_ to the cycle in which the latch became deterministic was identified. The procedure was performed concurrently for all 4000 latches. This resulted in the specification of the set of latches that can be scanned on each cycle of the reset sequence. Latches initialized by the sequence are not scanned until the cycle at which they become deterministic. Each of these scan specifications includes all the latches that were scanned on the previous cycle and may include new latches known to have become deterministic at that cycle.

The third and last step of the development process was to run the test on a fault free computer while scanning the deterministic latches at each cycle of the test. This information was used to generate the expected checksum values to be stored with the test. The latch values were saved in a data base to be used for identifying miscomparing latches.

CONCLUSIONS

This section identifies the problems encountered during the development and use of the Deterministic Reset Test and discusses how these could have been averted. It also points out the positive aspects of the testing technique that was used.

The following are the major problems encountered:

Different Machine Hardware at Various Customer Sites: This resulted from hardware design modifications, new fetures, and differing customer requirements. Because the test scans out up to 8000 latches over many machine cycles, it is very likely that a design change can invalidate the test operation. There is an intrinsic tradeoff between the test's ability to pin-point a failing latch and its capability to run on different machine levels. For example, consider comparing the operation of two machines, one with a performance enhancement change and one without it. This would result in some latches having different values at some cycle of the internal operations of the two machines due to the effect of the hardware change.

This problem of the test's dependence on the hardware design can be avoided if the test observes only errors indicating latches. These latches will have known values during fault free operation. They would not be affected by design changes. Any deviation would indicate fault detection. For this method to be effective, a large number of checkers must be included in the hardware design of the control sequencers as well as the data paths so that a failure can be detected as close to its source as possible.

Insufficient Reset by Hardware for Diagnostic Use: The test required too many operations to initialize all the latches of the machine. The reason for this was that the hardware reset of the machine only initialized its control logic. To initialize the data paths, the test had to perform specialized operations. This resulted in the large number of cycles in the test. Had the hardware reset initialized every latch of the machine, this problem would not have occurred and the development of a reset test would have been a simple process.

The following discusses the positive aspects of the testing technique.

Using a Diagnostic Processor: This provides a fault isolation capability that cannot be accomplished in any other way. It allows a test to directly observe the machine being tested in order to identify the cause of a failure accurately and to ensure that the fault situation can be re-created. In addition, the test's fault analysis and isolation will not destroy the fault indicating state of the tested machine. Such testing requires that the diagnostic processor support altering and displaying of the internal state of the tested machine, and the single cycling of the machine clocks.

The Channel Diagnostic Processor: Since this device is attached to the mainframe and uses the system clocks, it will provide a synchronous and repeatable I/O interface. The use of a micro-programmed processor enables the generation of non-standard or simulated error response for diagnostic purposes.

An Operating System Running in the Diagnostic Processor: This provides for a wide range of functions that can assist in the diagnosis of mainframe failures. These functions should include:

1. Formatted scanning of most of the internal machine latches. The display can associate a register name to its contents.

2. Altering of the states of significant registers and memories by keyboard commands.

3. Remote maintenance capability over standard telephone lines. This should include observations and alteration of a customer's mainframe from a field assistance center.

ACKNOWLEDGEMENTS

The authors would like to express their gratitude to those who contributed to the development of the test. They would like to thank Tom Hirst, Reed Larsen, and Robert Maier for their assistance in defining the sequence needed to initialize the machine. They would also like to thank Dave Anderson, Jeff Booth, Richard Bishop, Kim DeVaughn, Chacko Neroth, Jack Oneto, and Tom Wallters of the Software Engineering management for providing the needed resources for the development of this test, and thank Arun Shah for his excellent technical contributions to the test development.

REFERENCES

[1] "Checksum Test Methods", John P. Hayes, Proceedings of the 1976 International Symposium on Fault Tolerant computing, pp. 114-120, June, 1976.

[2] "Compact Testing with Compressed Data", Kenneth P. Parker, Proceedings of the 1976 International Symposium on Fault Tolerant Computing, pp. 93-98, June, 1976.

[3] "No. 1 ESS Maintenance Plan", R. W. Downing, J. S. Nowak, & L. S. Tuomenoksa, Bell System Technical Journal, February, 1977.

[4] "Automatic System Level Test Generation and Fault Location for Large Digital Systems", A. Yamada, N. Wakatsuki, T. Fukui, and S. Funatsu, Proceeding of the fifteenth annual Design Automation Conference, June, 1978.

[5] "LSSD Latch Configuration which requires fewer input changes for both Scan-in and Scan-out Operation", D. E. Lee, IBM technical disclosure bulletin, Vol 20, No. 1, pp. 265-267, June, 1977.

[6] "A Technique for Failure Isolation in a Large Computer System", A. Shah, Proceedings of the JEEE International Conference on Circuits and Computers, 1980. (To be published.)

[7] "Detailed Program Specifications for the Deterministic Reset Program", Amdahl internal document, part-number 805661-601 revision D, 04/26/76.

[8] "Detailed Program Specification for the Deterministic Reset Test", Amdahl internal document, part-number 806-73-601 revision B, 06/03/77.

[9] "Design Overview Specifications for the Incremental Domain Development System", Amdahl internal document, part-number 809034-600 revision A, 02/21/78.

[10] "Design Overview Specification for the Incremental Domain Development System Automation", Amdahl internal document, part-number 809077-601 revision A, 06/22/78.