(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 416 732 B1

(12) **EUROPEAN PATENT SPECIFICATION**

(45) Date of publication and mention of the grant of the patent:
30.12.1998 Bulletin 1998/53

(51) Int Cl.[6]: **G06F 1/24**, G06F 11/00, G06F 11/14

(21) Application number: **90308007.5**

(22) Date of filing: **20.07.1990**

(54) **Targeted resets in a data processor**

Gezielte Rücksetzungen in einem Datenprozessor

Remises à zéro sélectives dans un processeur de données

(84) Designated Contracting States:
AT BE CH DE DK ES FR GB GR IT LI LU NL SE

(30) Priority: **01.08.1989 US 388087**

(43) Date of publication of application:
13.03.1991 Bulletin 1991/11

(73) Proprietor: **DIGITAL EQUIPMENT CORPORATION**
Maynard, MA 01754 (US)

(72) Inventors:
- **Bruckert, William**
  Northboro, Massachusetts 01532 (US)
- **Kovalcin, David**
  Grafton, Massachusetts 01519 (US)
- **Bissett, Thomas D.**
  Derry, New Hampshire 03038 (US)
- **Munzer, John**
  Brookline, Massachusetts 02146 (US)
- **Norcross, Mitchell**
  Nashua, New Hampshire 03062 (US)

(74) Representative: **Goodman, Christopher et al**
Eric Potter Clarkson,
Park View House,
58 The Ropewalk
Nottingham NG1 5DD (GB)

(56) References cited:
EP-A- 0 077 154
EP-A- 0 306 244
US-A- 4 580 232
US-A- 4 757 442

- **IBM TECHNICAL DISCLOSURE BULLETIN vol. 29, no. 8 , January 1987 , NEW YORK US pages 3562 - 3563 'PROGRAMMABLE SYSTEM AND POWER'**

EP 0 416 732 B1

Description

I. BACKGROUND OF THE INVENTION

The present invention relates to the field of resetting a data processor and, more particularly, to the field of managing different classes of resets in a data processor.

All data processing systems need the capability of resetting under certain conditions, such as during power up or when certain errors occur. Without resets there would be no way to set the data processing system into a known state either to begin initialization routines or to begin error recovery routines.

The problem with resets, however, is that they have wide-ranging effects. In general, resets disrupt the normal flow of instruction execution and may cause a loss of data or information. Sometimes such drastic action is required to prevent more serious problems, but often the effect of the resets is worse than the condition which caused the resets.

Another problem with resets in conventional machines is that they are not localized. In other words, an entire data processing system is reset when only a portion needs to be. This is particularly a problem in systems employing multiple processors such as for fault-tolerant applications. In such systems, an error in one of the processors can propagate to the other processors and bring the entire system to a halt. If the originating processor was in error in generating resets, then the effect is to cause an unnecessary halt in execution.

It would therefore be advantageous to design a system in which the resets are matched to the conditions which generated the reset.

It would also be advantageous for such a system to have several classes of resets with different effects.

It would be additionally advantageous if, in a multiple processor data processing system, the resets in one of the processors did not automatically propagate to the other processors.

Additional advantages of this invention will be set forth in part in the description which follows and in part will be obvious from that description or may be learned by practising the invention. The advantages may be realized by the methods and apparatus particularly pointed in the appended claims.

US Patent 4,580,232 to Dungan et al, teaches designation of one of the processors in a system as a master processor, in the event of a software crash, to reset the other processors. A reset signal is automatically conveyed to the other processors by the master processor to restore the other processors back to normal operation.

IBM Technical Disclosure Bulletin, Vol. 29, No. 8, January 1987 teaches a technique that allows a program in the processor to invoke resets equivalent to system reset and power-on reset for unattended environments.

European published Patent Application No. 0 306 244 A2 teaches a fault tolerant computer system with fault isolation and repair. Error checking devices detect presence of errors in the CPU. Error storage devices are coupled to transaction data storage devices and error checking devices for stopping storage of additional messages in transaction data storage devices in the event of detected errors.

II. SUMMARY OF THE INVENTION

The present invention, in its broad form, resides in a method and system of resetting a data processing system without altering the sequence of instructions of the steps executed by the data processing system, as recited in claims 1 and 11 respectively.

III. BRIEF DESCRIPTION OF THE DRAWINGS

The accompanying drawings, which are incorporated in and which constitute a part of this specification illustrate one embodiment of the invention and, together with the description of the invention, explain the principles of the invention.

Fig. 1 is a block diagram of a preferred embodiment of fault tolerant computer system which practices the present invention;

Fig. 2 is an illustration of the physical hardware containing the fault tolerant computer system in Fig. 1;

Fig. 3 is a block diagram of the CPU module shown in the fault tolerant computer system shown in Fig. 1;

Fig. 4 is a block diagram of an interconnected CPU module and I/O module for the computer system shown in Fig. 1;

Fig. 5 is a block diagram of a memory module for the fault tolerant computer system shown in Fig. 1;

Fig. 6 is a detailed diagram of the elements of the control logic in the memory module shown in Fig. 5;

Fig. 7 is a block diagram of portions of the primary memory controller of the CPU module shown in Fig. 3;

Fig. 8 is a block diagram of the DMA engine in the primary memory controller of the CPU module of Fig. 3;

Fig. 9 is a diagram of error processing circuitry in the primary memory controller of the CPU module of Fig. 3;

Fig. 10 is a drawing of some of the registers of the cross-link in the CPU module shown in Fig. 3;

Fig. 11 is a block diagram of the elements which route control signals in the cross-links of the CPU module shown in Fig. 3;

Fig. 12 is a block diagram of the elements which route data and address signals in the primary cross-link of the CPU module shown in Fig. 3;

Fig. 13 is a state diagram showing the states for the cross-link of the CPU module shown in Fig. 3;

Fig. 14 is a block diagram of the timing system for the fault tolerant computer system of Fig. 1;
Fig. 15 is a timing diagram for the clock signals generated by the timing system in Fig. 14;
Fig. 16 is a detailed diagram of a phase detector for the timing system shown in Fig. 14;
Fig. 17 is a block diagram of an I/O module for the computer system of Fig. 1;
Fig. 18 is a block diagram of the firewall element in the I/O module shown in Fig. 17;
Fig. 19 is a detailed diagram of the elements of the cross-link pathway for the computer system of Fig. 1;
Figs. 20A-20E are data flow diagrams for the computer system in Fig. 1;
Fig. 21 is a block diagram of zone 20 showing the routing of reset signals;
Fig. 22 is a block diagram of the components involved in resets in the CPU module shown in Fig. 3; and
Fig. 23 is a diagram of clock reset circuitry.

IV. DESCRIPTION OF THE PREFERRED EMBODIMENT

Reference will now be made in detail to a presently preferred embodiment of the invention, an example of which is illustrated in the accompanying drawings.

A. SYSTEM DESCRIPTION

Fig. 1 is a block diagram of a fault tolerant computer system 10 in accordance with the present invention. Fault tolerant computer system 10 includes duplicate systems, called zones. In the normal mode, the two zones 11 and 11' operate simultaneously. The duplication ensures that there is no single point of failure and that a single error or fault in one of the zones 11 or 11' will not disable computer system 10. Furthermore, all such faults can be corrected by disabling or ignoring the device or element which caused the fault. Zones 11 and 11' are shown in Fig. 1 as respectively including duplicate processing systems 20 and 20'. The duality, however, goes beyond the processing system.

Fig. 2 contains an illustration of the physical hardware of fault tolerant computer system 10 and graphically illustrates the duplication of the systems. Each zone 11 and 11' is housed in a different cabinet 12 and 12', respectively. Cabinet 12 includes battery 13, power regulator 14, cooling fans 16, and AC input 17. Cabinet 12' includes separate elements corresponding to elements 13, 14, 16 and 17 of cabinet 12.

As explained in greater detail below, processing systems 20 and 20' include several modules interconnected by backplanes. If a module contains a fault or error, that module may be removed and replaced without disabling computing system 10. This is because processing systems 20 and 20' are physically separate, have separate backplanes into which the modules are plugged, and can operate independently of each other. Thus modules can be removed from and plugged into the backplane of one processing system while the other processing system continues to operate.

In the preferred embodiment, the duplicate processing systems 20 and 20' are identical and contain identical modules. Thus, only processing system 20 will be described completely with the understanding that processing system 20' operates equivalently.

Processing system 20 includes CPU module 30 which is shown in greater detail in Figs. 3 and 4. CPU module 30 is interconnected with CPU module 30' in processing system 20' by a cross-link pathway 25 which is described in greater detail below. Cross-link pathway 25 provides data transmission paths between processing systems 20 and 20' and carries timing signals to ensure that processing systems 20 and 20' operate synchronously.

Processing system 20 also includes I/O modules 100, 110, and 120. I/O modules 100, 110, 120, 100', 110' and 120' are independent devices. I/O module 100 is shown in greater detail in Figs. 1, 4, and 17. Although multiple I/O modules are shown, duplication of such modules is not a requirement of the system. Without such duplication, however, some degree of fault tolerance will be lost.

Each of the I/O modules 100, 110 and 120 is connected to CPU module 30 by dual rail module interconnects 130 and 132. Module interconnects 130 and 132 serve as the I/O interconnect and are routed across the backplane for processing system 20. For purposes of this application, the data pathway including CPU 40, memory controller 70, cross-link 90 and module interconnect 130 is considered as one rail, and the data pathway including CPU 50, memory controller 75, cross-link 95, and module interconnect 132 is considered as another rail. During proper operation, the data on both rails is the same.

B. FAULT TOLERANT SYSTEM PHILOSOPHY

Fault tolerant computer system 10 does not have a single point of failure because each element is duplicated. Processing systems 20 and 20' are each a fail stop processing system which means that those systems can detect faults or errors in the subsystems and prevent uncontrolled propagation of such faults and errors to other subsystems, but they have a single point of failure because the elements in each processing system are not duplicated.

The two fail stop processing systems 20 and 20' are interconnected by certain elements operating in a defined manner to form a fail safe system. In the fail safe system embodied as fault tolerant computer system 10, the entire computer system can continue processing even if one of the fail stop processing systems 20 and 20' is faulting.

The two fail stop processing systems 20 and 20' are considered to operate in lockstep synchronism because CPUs 40, 50, 40' and 50' operate in such synchronism. There are three significant exceptions. The first is at initialization when a bootstrapping technique brings both processors into synchronism. The second exception is when the processing systems 20 and 20' operate independently (asynchronously) on two different workloads. The third exception occurs when certain errors arise in processing systems 20 and 20'. In this last exception, the CPU and memory elements in one of the processing systems is disabled, thereby ending synchronous operation.

When the system is running in lockstep I/O, only one I/O device is being accessed at any one time. All four CPUs 40, 50, 40' and 50', however, would receive the same data from that I/O device at substantially the same time. In the following discussion, it will be understood that lockstep synchronization of processing systems means that only one I/O module is being accessed.

The synchronism of duplicate processing systems 20 and 20' is implemented by treating each system as a deterministic machine which, starting in the same known state and upon receipt of the same inputs, will always enter the same machine states and produce the same results in the absence of error. Processing systems 20 and 20' are configured identically, receive the same inputs, and therefore pass through the same states. Thus, as long as both processors operate synchronously, they should produce the same results and enter the same state. If the processing systems are not in the same state or produce different results, it is assumed that one of the processing systems 20 and 20' has faulted. The source of the fault must then be isolated in order to take corrective action, such as disabling the faulting module.

Error detection generally involves overhead in the form of additional processing time or logic. To minimize such overhead, a system should check for errors as infrequently as possible consistent with fault tolerant operation. At the very least, error checking must occur before data is outputted from CPU modules 30 and 30'. Otherwise, internal processing errors may cause improper operation in external systems, like a nuclear reactor, which is the condition that fault tolerant systems are designed to prevent.

There are reasons for additional error checking. For example, to isolate faults or errors it is desirable to check the data received by CPU modules 30 and 30' prior to storage or use. Otherwise, when erroneous stored data is later accessed and additional errors result, it becomes difficult or impossible to find the original source of errors, especially when the erroneous data has been stored for some time. The passage of time as well as subsequent processing of the erroneous data may destroy any trail back to the source of the error.

"Error latency," which refers to the amount of time an error is stored prior to detection, may cause later problems as well. For example, a seldom-used routine may uncover a latent error when the computer system is already operating with diminished capacity due to a previous error. When the computer system has diminished capacity, the latent error may cause the system to crash.

Furthermore, it is desirable in the dual rail systems of processing systems 20 and 20' to check for errors prior to transferring data to single rail systems, such as a shared resource like memory. This is because there are no longer two independent sources of data after such transfers, and if any error in the single rail system is later detected, then error tracing becomes difficult if not impossible. The preferred method of error handling is set forth in Application No. 90308000.0 filed this same date entitled, "Software Error Handling", and published as EP-0415545.

C. MODULE DESCRIPTION

1. CPU Module

The elements of CPU module 30 which appear in Fig. 1 are shown in greater detail in Figs. 3 and 4. Fig. 3 is a block diagram of the CPU module, and Fig. 4 shows block diagrams of CPU module 30 and I/O module 100 as well as their interconnections. Only CPU module 30 will be described since the operation of and the elements included in CPU modules 30 and 30' are generally the same.

CPU module 30 contains dual CPUs 40 and 50. CPUs 40 and 50 can be standard central processing units known to persons of ordinary skill. In the preferred embodiment, CPUs 40 and 50 are VAX microprocessors manufactured by Digital Equipment Corporation, the assignee of this application.

Associated with CPUs 40 and 50 are cache memories 42 and 52, respectively, which are standard cache RAMs of sufficient memory size for the CPUs. In the preferred embodiment, the cache RAM is 4K x 64 bits. It is not necessary for the present invention to have a cache RAM, however.

2. Memory Module

Preferably, CPU's 40 and 50 can share up to four memory modules 60. Fig. 5 is a block diagram of one memory module 60 shown connected to CPU module 30.

During memory transfer cycles, status register transfer cycles, and EEPROM transfer cycles, each memory module 60 transfers data to and from primary memory controller 70 via a bidirectional data bus 85. Each memory module 60 also receives address, control, timing, and ECC signals from memory controllers 70 and 75 via buses 80 and 82, respectively. The address signals on buses 80 and 82 include board, bank, and row and column address signals that identify the mem-

ory board, bank, and row and column address involved in the data transfer.

As shown in Fig. 5, each memory module 60 includes a memory array 600. Each memory array 600 is a standard RAM in which the DRAMs are organized into eight banks of memory. In the preferred embodiment, fast page mode type DRAMs are used.

Memory module 60 also includes control logic 610, data transceivers/registers 620, memory drivers 630, and an EEPROM 640. Data transceivers/receivers 620 provide a data buffer and data interface for transferring data between memory array 600 and the bidirectional data lines of data bus 85. Memory drivers 630 distribute row and column address signals and control signals from control logic 610 to each bank in memory array 600 to enable transfer of a longword of data and its corresponding ECC signals to or from the memory bank selected by the memory board and bank address signals.

EEPROM 640, which can be any type of NVRAM (nonvolatile RAM), stores memory error data for off-line repair and configuration data, such as module size. When the memory module is removed after a fault, stored data is extracted from EEPROM 640 to determine the cause of the fault. EEPROM 640 is addressed via row address lines from drivers 630 and by EEPROM control signals from control logic 610. EEPROM 640 transfers eight bits of data to and from a thirty-two bit internal memory data bus 645.

Control logic 610 routes address signals to the elements of memory module 60 and generates internal timing and control signals. As shown in greater detail in Fig. 6, control logic 610 includes a primary/mirror designator circuit 612.

Primary/mirror designator circuit 612 receives two sets of memory board address, bank address, row and column address, cycle type, and cycle timing signals from memory controllers 70 and 75 on buses 80 and 82, and also transfers two sets of ECC signals to or from the memory controllers on buses 80 and 82. Transceivers/registers in designator 612 provide a buffer and interface for transferring these signals to and from memory buses 80 and 82. A primary/mirror multiplexer bit stored in status registers 618 indicates which one of memory controllers 70 and 75 is designated as the primary memory controller and which is designated as the mirror memory controller, and a primary/mirror multiplexer signal is provided from status registers 618 to designator 612.

Primary/mirror designator 612 provides two sets of signals for distribution in control logic 610. One set of signals includes designated primary memory board address, bank address, row and column address, cycle type, cycle timing, and ECC signals. The other set of signals includes designated mirror memory board address, bank address, row and column address, cycle type, cycle timing, and ECC signals. The primary/mirror multiplexer signal is used by designator 612 to select whether the signals on buses 80 and 82 will be respectively routed to the lines for carrying designated primary signals and to the lines for carrying designated mirror signals, or vice-versa.

A number of time division multiplexed bidirectional lines are included in buses 80 and 82. At certain times after the beginning of memory transfer cycles, status register transfer cycles, and EEPROM transfer cycles, ECC signals corresponding to data on data bus 85 are placed on these time division multiplexed bidirectional lines. If the transfer cycle is a write cycle, memory module 60 receives data and ECC signals from the memory controllers. If the transfer cycle is a read cycle, memory module 60 transmits data and ECC signals to the memory controllers. At other times during transfer cycles, address, control, and timing signals are received by memory module 60 on the time division multiplexed bidirectional lines. Preferably, at the beginning of memory transfer cycles, status register transfer cycles, and EEPROM transfer cycles, memory controllers 70 and 75 transmit memory board address, bank address, and cycle type signals on these timeshared lines to each memory module 60.

Preferably, row address signals and column address signals are multiplexed on the same row and column address lines during transfer cycles. First, a row address is provided to memory module 60 by the memory controllers, followed by a column address about sixty nanoseconds later.

A sequencer 616 receives as inputs a system clock signal and a reset signal from CPU module 30, and receives the designated primary cycle timing, designated primary cycle type, designated mirror cycle timing, and designated mirror cycle type signals from the transceivers/registers in designator 612.

Sequencer 616 is a ring counter with associated steering logic that generates and distributes a number of control and sequence timing signals for the memory module that are needed in order to execute the various types of cycles. The control and sequence timing signals are generated from the system clock signals, the designated primary cycle timing signals, and the designated primary cycle type signals.

Sequencer 616 also generates a duplicate set of sequence timing signals from the system clock signals, the designated mirror cycle timing signals, and the designated mirror cycle type signals. These duplicate sequence timing signals are used for error checking. For data transfers of multi-long words of data to and from memory module 60 in a fast page mode, each set of column addresses starting with the first set is followed by the next column address 120 nanoseconds later, and each long word of data is moved across bus 85 120 nanoseconds after the previous long word of data.

Sequencer 616 also generates tx/rx register control signals. The tx/rx register control signals are provided to control the operation of data transceivers/registers 620 and the transceivers/registers in designator 612. The direction of data flow is determined by the steering

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.