Diagnostic and debugging arrangement for a data processing system.

Priority: 28.12.79 US 107992

Date of publication of application: 08.07.81 Bulletin 81/27

Publication of the grant of the patent: 22.08.84 Bulletin 84/34

Designated Contracting States: DE FR GB

References cited:
DE-A-2 625 183
GB-A-1 536 147
US-A-3 814 920
US-A-3 964 088
US-A-4 023 142


Proprietor: International Business Machines Corporation
Old Orchard Road
Armonk, N.Y. 10504 (US)

Inventor: Bantz, David Frederick
65 Ridgewood Terrace
Chappaqua New York 10504 (US)
Inventor: Evangelisti, Carlo John
Orchard Road
Jefferson Valley New York 10535 (US)
Inventor: Nelson, Robert Arthur
Kingston Road RD
Katoh New York 10536 (US)

Representative: Möhlen, Wolfgang C., Dipl.-Ing.
Säumerstrasse 4
CH-8803 Rüschlikon (CH)

Note: Within nine months from the publication of the mention of the grant of the European patent, any person may give notice to the European Patent Office of opposition to the European patent granted. Notice of opposition shall be filed in a written reasoned statement. It shall not be deemed to have been filed until the opposition fee has been paid. (Art. 99(1) European patent convention).
Description

Present invention is concerned with a diagnostic and debugging arrangement for a data processing system which comprises a host computer incorporating a level-sensitive scan register and having interface lines for transfer of level-sensitive scan data and for sense and control signals. In particular, the invention is concerned with such a diagnostic and debugging arrangement comprising a diagnostic computer and an interface unit interposed between said host computer and said diagnostic computer in data exchange relationship with both, and wherein said interface unit includes register means for receiving state data from the level-sensitive scan register of the host computer to be transferred to said diagnostic computer, and for storing state data from the diagnostic computer to be stored in predetermined locations in the level-sensitive scan register of said host computer, and further includes control means actuable by said diagnostic computer to cause state data to be selectively transferred between said host computer and said diagnostic computer via said register means, and wherein said diagnostic computer includes control means for effecting the execution of predetermined diagnostic programs which require the access of state data from and the transfer of state data to the host computer.

Electronic data processing systems, almost from their inception, have been provided with various types of error checking and diagnostic circuits and techniques for supervising the arithmetic and logical operations carried out within the same. Probably the best known, generate an additional parity bit for each data word. By means of the parity bit the number of bits within each data word which have a given binary value (e.g. “1”) is caused to be either even or odd. The parity of the data word can be newly formed from one processing step to the next and be compared to the original one.

More sophisticated types of error detection mechanisms based on Hamming codes have been utilized in the past in systems where more efficient error detection capability is required. With the use of Hamming codes the particular portion of a fixed length data word in error may be accurately diagnosed. Depending upon the amount of additional redundant data the system designer is willing to carry, errors may be diagnosed down to a single bit position utilizing Hamming techniques.

With both, parity bit techniques and Hamming code techniques that result of an “error” indication is simply that an error has been detected by the system and at this point some sort of diagnostic routine must be entered to determine the nature of the error and what must be done to correct same. The simplest form of corrective measure is normally a “retry” wherein the system causes the erroneous operation to be repeated in hopes that the error is transient in nature. If a particular system is sufficiently modular and an error is found to be non-transient, it is possible to often switch in a standby module for one that has been found to be defective. Still other corrective measures used in very high reliability systems include Triple Modular Redundancy (TMR). In TMR systems critical portions of the system are redundantly designed. By this is meant that three embodiments of each unit are included in the overall system, and voting circuits located at critical points test whether at least two out of three of the results are the same. When one unit is found to deviate, an error is indicated; however, the system may continue to operate utilizing the majority output from the voting circuit. It is noted that even with TMR systems only the existence of an error is normally detected and not its specific cause.

With most modern day computing systems, once an error is detected and an appropriate signal is provided, human personnel must intervene directly and normally cause special diagnostic procedures and hardware to interrupt the system operation to exhaustively diagnose that portion of the system which has been found to be defective. Stated very generally, most such diagnostic routines involve the exhaustive testing of the faulty modules with particular diagnostic data test patterns which are specifically designed to test the hardware until the permanent failure is isolated.

Such diagnostic procedures are extremely costly since the operation of the system must be totally interrupted for extended periods of time and further intervention of human operators or engineers is required. A recent development in the field of computer diagnostics has been the concept described as Level Sensitive Scan Design (LSSD) wherein the logical circuits within a computing system are provided with a special test register which can be directly accessed by suitable diagnostic programs so that as a test pattern is being passed through the system the condition of various circuits may be continuously monitored.

For a detailed description of the LSSD design concept, reference should be made to the first three US patents mentioned in the Background Art section of present specification, and also to the article “A Logic Design Structure for LSI Testability” by E. B. Eichelberger and T. W. Williams, Proceedings of the “14th Design Automation Conference”, New Orleans, 1977, pp. 462—468.

Another problem area with electronic data processing systems and particularly with sophisticated and complex modern computers is that of debugging or detecting errors in system software. Based on a recognition that system software errors are often very closely related to hardware errors, a current approach in the industry is one wherein both hardware and system software are debugged concurrently. The machine environ-
ment has become more complex with virtual memory, concurrent channels, imprecise interrupts, pipelined operations and interruptable instructions. The problem of debugging system software in this environment is difficult and the usual debugging facilities utilizing one another's instruction traces are insufficient due to the complex interactions between the channels, interrupts, and the instruction execution. For example, in the IBM System/370 model 168, the Move Characters Long Instruction is interruptable in order to reduce interrupt latency. If the interrupt routine modifies the operands before returning, the effect of the instruction will be altered. These facts will not be reflected by a single-instruction trace facility. There is also no way to cause an interrupt to occur at a precise point during instruction execution in order to investigate these facts. What is required, in effect, is a machine that is controllable at a much finer level by the person debugging same.

It has been known to routinely use a small processor to gather and reduce information from a machine under test. When the debugging processor is not attached, the host computer runs full speed. Hardware is designed to allow easy access for the entire machine state to facilitate the debugging via the above referenced LSSD design rules. Such previously utilized debugging techniques only provide a means for diagnosing hardware faults, however, neither hardware nor software facilities are provided to support debugging of an arbitrary program on the main computer under arbitrary machine states. It will be readily recognized that there is a great need in the electronic data processing industry for continually improved methods for both diagnosing and debugging the hardware and the system software of the host computer. It is also highly desirable that such diagnostic and debugging systems require a minimum of direct human intervention as a provision.

Background art

The following three US patents are illustrative of the LSSD testing arrangements and organizations and should be referred to for detailed descriptions of their underlying principles.


The following further three US patents are cited as being generally exemplary of the system diagnostic art, particularly U.S. Patent No. 3,786,430 entitled "Data Processing System Including a Small Auxiliary Processor for Overcoming the Effects of Faulty Hardware", granted January 15, 1974. This patent discloses an electronic data processing system comprising a main processor and an auxiliary processor and is designed to minimize the effects of hardware failure by intercepting the function of the main processor in the case of a hardware error. Although the auxiliary processor is in some sense a monitor, it is better described as an error processing system. It is linked to the data registers and function decoders of the main processor, and designed so that it can accept the necessary input data in order to simulate the operation of the main processor in the case of hardware failure. It transfers its output data to the main processor once the simulated function is complete.

U.S. Patent 3,585,599 entitled "Universal System Service Adapter", granted June 15, 1971. This patent discloses a universal adapter which serves as an interface between its host computer and external equipment, another processor, for purposes of monitoring and testing the host. The adapter is designed to receive diagnostic test control information from the external source for transmittal to the host system and to transmit system status information from the host to the external source. Testing is, however, limited to only certain hardware functions, with test responses being of a pass/fail nature only. The transmitted information, i.e., from host to external equipment is similarly limited to system status log messages.

U.S. Patent 3,825,901 entitled "Integrated Diagnostic Tool", granted July 23, 1974. This patent discloses an integrated diagnostic tool which is an enhancement of the adapter described in the above mentioned US Patent (3,585,599), which permits testing and monitoring while the system is running at normal speed.

None of the above diagnostic system patents discloses a combined diagnostic and debugging arrangement for a data processing system allowing detailed and flexible interaction and selective transfers between a host and an auxiliary processor, and none of them makes any suggestion of such a diagnostic tool specifically adapted and tailored to be used with the LSSD design concept.

In GB—A—1 536 147 a data processing system is disclosed having arrangements for obtaining access to the internal registers of such a system, e.g., for diagnostic purposes. The system comprises a central processor including internal registers, and an auxiliary processor which can be used for diagnostics. The registers are formed from shift register components which are interconnected in series to form a plurality of loops. By selecting specific loops, data contained in selected internal registers can be serially shifted into a test register where they can be read by the auxiliary processor, or test data can be placed into the test register from where they can be serially shifted into selected
internal registers which are interconnected in selected loops. The disclosed arrangement requires that the auxiliary processor is immediately involved in the selection operations during transfer of test data from internal registers to the test register and vice versa, thus is not available for processing other test data during that time. Furthermore, the central processor’s internal registers are directly involved in the selection operations in varying loop configurations.

It is a primary object of the present invention to provide an improved diagnostic and debugging arrangement for use with highly sophisticated modern electronic data processing systems.

It is a further object of the invention to provide such a system for use with host CPUs designed in accordance with LSSD design principles.

It is yet another object of the invention to provide such a diagnostic and testing facility which is capable of three modes of operation. These are run continuous, stop at the end of each instruction, and stop at the end of each machine cycle.

It is another particular object of present invention to provide a diagnostic and debugging arrangement in which state data can be selectively extracted from or inserted into consecutive data patterns accessible through LSSD register means, independently of the host computer and diagnostic computer to avoid unnecessary interference with their operation for these selections.

It is a further object of the invention to provide such diagnostic and debugging arrangement in which the command signals for independent control of state data transfer operations can be derived in the diagnostic computer in a simple manner and with a minimum of additional circuitry.

The arrangement for achieving these objects is characterized in claim 1. Preferred further details of the invention are represented in claims 2, 3 and 4.

Principles of the invention and details of an embodiment will be described in the sequel in connection with drawings.

Description of the drawings

Figure 1 is an organizational block diagram showing the overall architecture of the present system including the host computer, the interface and the diagnostic computer.

Figure 2 is a functional block diagram of the diagnostic computer.

Figure 3 is a functional block diagram of the interface unit of Figure 1.

Figure 4 comprises a functional block diagram of that portion of the architecture of the host computer, including the LSSD register and various control lines, requisite for interfacing with the present system architecture.

Figure 5 comprises a combined functional and logical schematic diagram of the command control unit located in the diagnostic computer.

Figure 6 illustrates the address format of the instructions produced by the D-CPU to be utilized by the command control unit of Figure 5.

Figure 7 comprises a combined functional and logical schematic diagram of the shift control unit located in the interface unit of Figure 3.

Figure 8 comprises a combined functional and logical schematic of the start/stop control unit of the interface unit of Figure 3.

Figure 9 comprises a combined functional block and logical schematic diagram of the LSSD control unit of the interface unit of Figure 3.

Figure 10 comprises a functional block diagram of an “early selection” enhancement feature for the present system.

General principles and conditions

The presently disclosed diagnostic/debug arrangement including its own unique interface unit is specifically designed and intended for use with a host computer embodying the LSSD design concept throughout. It should be noted that the present invention is intended to cover only the overall architecture of such a system and not any specific diagnostic programs or routines utilized by the diagnostic computer once an error or some other anomalous situation occurs. Thus the presently disclosed architecture discloses an architectural framework providing all the necessary controls for interfacing with a host computer’s interrupt system for the purpose of transferring requisite LSSD state data and for signaling the host system when it is to continue running.

Also disclosed herein are the necessary registers and controls for saving LSSD state data from the host system and returning said state data back to the LSSD registers of the host system in either original or altered form.

According to an additional aspect of the presently disclosed system, controls are provided for examining only a specified portion of the state data obtained from the host system’s LSSD registers which results in a minimizing of interrupt time for the host CPU.

It will be noted from the following description that the only physical host lines required by the present system’s interface are a line for obtaining “end of cycle” and “end of instruction” data signals. Also a “continue operation” (Run H-CUP) line must be provided for telling the host CPU to continue its normal operating mode or remain interrupted, which signal is produced by the present system. Additionally, two lines must be provided for accessing LSSD state data from the machine and for returning LSSD state data to the machine. Further, an LSSD register shift control line must be provided by the present system for control of
the shifting of data into and out of the host computer's LSSD registers. It will be appreciated that such control lines are readily available at the external interface of an LSSD designed machine.

It will also be readily understood by those skilled in the art that the herein disclosed diagnostic/debug arrangement provides an architecture which will allow a variety of hardware, error checking as well as software debugging functions to be performed. By providing a finesse of control in the present system to the point where the host computer may be interrupted at the completion of every machine cycle, extremely powerful and exhaustive testing of the host system is possible.

The present diagnostic/debug arrangement allows the machine state and the memory of the host CPU to be captured at the end of an instruction or at the end of a cycle as stated above. It further allows selective reading and writing of the memory state by conditioning the host computer with state information that will cause words from or to memory to be transferred to the diagnostic computer. The system allows control over interruption, channel activity and address translation and provides a diagnostic CPU with its own memory to perform diagnostic and debug functions. These and other objects, features and advantages of the system will be apparent from the subsequent description of the preferred embodiment of the system.

Before proceeding with a detailed description of the present system there will follow a very brief description of the principles of Level Sensitive Scan Design (LSSD) concepts to more clearly place the present invention in perspective. The common main thrust of the LSSD design concept is to prescribe a built-in capability for every replaceable module, whereby the entire logic state of the unit, when under test, can be explicitly set and/or examined through exercising certain input/output (I/O) procedures at a limited number of I/O terminals. This requirement is implementable by imparting a shift-register capability to every one of the logic system latches in the unit and thereupon organizing these shift register latches (SRL) into one or more shift register data channels with their terminal stages accessible to the outside world (via the interface).

Details of operations using the SRL facility for various aspects of the testing purposes are given in most of the aforementioned patents. Particular reference may be made to Figure 8 of US Patent 3,761,695 and Figures 7, 8 and 9 of US Patent 3,784,907. Stated very briefly, the LSSD approach comprises a test operation wherein certain desired logic-test patterns are serially inputted and shifted to the appropriate latch locations when the unit is operated in the "shift mode", so to speak, (i.e., by withholding the system clock excitations and turning on the shifting clock to the unit). When this is done, the latch states will provide the desired stimuli for the testing of the related logic nets. Now, propagate the test patterns through the nets by executing one or more steps of the "Function Mode" operation (i.e., by exercising one or more system clock excitations). The response pattern of the logic networks to the applied stimuli is now captured by the system latches, in a known manner depending on certain details of hardware design, often replacing the original inputted test patterns. Then the system reverts to the shift-mode operation, outputting the response patterns for examination and comparison with standard patterns which should be present if the circuitry has operated properly.

It will be appreciated that the I/O terminals referred to in the above description would be replaced by the interface unit of the present diagnostic/debug arrangement. It is further assumed that all of the LSSD registers in the host CPU could be connected together in a predetermined fashion to in effect form one extremely long continuous shift register. Alternatively, the latches could be organized into multiple shorter shift registers, selected by known means by the diagnostic computer before specific state information is accessed. For the purposes of the present system it is assumed that this register would be no longer than 32 words each containing 32 bits. This will be apparent from the subsequent description. An extension of the present design to accommodate a longer LSSD register is set forth.

Description of an embodiment

There will now follow a general description of the operation of the present system which should be read with reference to the included figures. It will be noted that Figure 1 comprises, in essence, an organizational drawing of the present diagnostic/debug arrangement to a host. Figures 2, 3 and 4 comprise more detailed functional block diagrams of the hardware of each of the blocks of Figure 1. Similarly, Figures 5, 7, 8 and 9 are combined functional block and logical schematic diagrams of the primary functional units of Figures 2 and 3.

It will also be noted in referring to the figures that all of the lines and cables interconnecting the various units are shown as such with the exception of Figure 1. It will be appreciated that each of the single lines will carry a signal representing, i.e., a "1" or a "0", which would be represented, for example, by either a zero voltage appearing on the line ("0") or some predetermined positive or negative voltage differing from zero ("1"). Similarly a pulse might periodically appear on a line such as with the shifting pulse, which emanates from the interface unit of Figure 3 to control the LSSD register within the host computer. The cables are marked to indicate the number of lines in same. Thus, referring to Figure 2, the data bus used for the transfer of 32 bit data words between the LSSD control block located in the
interface unit on Figure 3, and the diagnostic computer contains 32 lines for transferring a data word in parallel between these units on command. It will be similarly noted that the LSSD Out and LSSD In lines utilized for transferring state data between the LSSD registers in the host computer and the registers within the LSSD control block of the interface unit are single lines. This configuration is necessary since, as will be remembered, all of the latches making up the total LSSD register in CPUs utilizing the LSSD design concept are in fact shift registers which can only be serially shifted (and read out) a bit at a time.

Also in Figure 2 within the diagnostic computer, the data bus is shown as containing 32 bits as it emanates from the interface unit and 54 bits as it passes into the command control block in the D-CPU. This is because only the state data words of 32 bits are being transferred between the interface unit and the diagnostic computer, and no address data is included therewith as the storage locations or addresses for the data coming from and going to the interface unit are specified by the instructions contained within the D-CPU. However, within the diagnostic computer the bus format is considered to have 54 bits of which 32 bits are data bits and 22 bits are devoted to address plus a read/write bit. As will be apparent from the subsequent description of Figure 6 the address per se utilizes 21 bits and a read/write bit is a single bit.

Referring now specifically to the drawings, Figure 1 shows an overall view of the diagnostic/debug arrangement including a diagnostic computer (D-computer) 11 connected to a host computer (H-computer) 12 via an interface unit 13. Any one of seven commands may be issued by the D-computer. These are carried to the interface via the command lines 15. In the interface, commands are interpreted and the state of the H-computer is selectively retrieved or selectively changed via the LSSD lines 16. In addition the H-computer signals the interface, via the host sense and control lines 17, when it has stopped and the interface will in turn run the H-computer or keep it stopped. The mode of operating the host is established by three of the seven commands. The three modes are “run continuous”, “stop at end of cycle” and “stop at end of instruction”. Communication between the interface and the D-computer, including memory, is via the data and control lines 18. The interface can transfer state data to either the H-computer or the D-computer. State data is intended to refer to the specific data content obtained from or to be stored in the LSSD latches of the H-computer where it specifies the state of the operational latches of said H-computer. Additionally, an interrupt, via the interrupt line 19, is sent to the D-computer whenever the H-computer has stopped. This interrupt is used by the D-computer to send or receive state data and start the host CPU (H-CPU).

Figures 2, 3 and 4 respectively describe in more detail the D-computer 11, the interface 13 and the H-computer 12.

Figure 2 shows that the D-computer 11 consists of a CPU (D-CPU) 21, a command control unit 22 and a direct memory access (DMA) unit 23 for allowing the interface and the D-CPU to access memory 24. The DMA is a conventional direct memory access such as AM9517 Multimode DMA Controller manufactured by Advanced Micro Devices or a PPS-8 DMA Controller manufactured by Rockwell Inc. It contains, as shown, an address and count register for each of two channels. These can be loaded from the D-CPU. When data is read or written into the memory via a channel, the address and count for that channel are incremented. Thus, when the next access is made, the registers have correct values. In this configuration the respective function of the two channels are: channel 1 to read from the memory of the D-computer and channel 2 to write into the memory of the D-computer. The DMA emits a signal when the data bus 25 is available for reading (strobe signal line 26) and a signal for writing (enable signal line 27). A typical operation of the DMA is to receive a request to either read or write memory. The DMA sends a request signal to the D-CPU and when the acknowledgement signal is received, the data bus is available and appropriate strobe and enable signals are issued to be used by the requester of either channel 1 or channel 2. In the present application of the DMA, the D-CPU is stopped when data is transferred to and from the Interface. The “run D-CPU” signal on line 28 from the command control unit 23 accomplishes this.

Command control unit 22 constantly monitors the address portion of the data/address/R/W bus. It detects an address of a given range, interprets that address as a command, and issues a signal on the command lines 15 to the interface. At the same time it issues a signal on the “run D-CPU” line 28 to control whether the D-CPU should be running or stopped. There are two commands (send state and receive state) which require a long transfer of data. When this transfer of data is complete, the transfer complete signal is received from the Interface on line 29. This signal is used to resume operation of the D-CPU. When the H-computer has stopped, the interrupt signal is received on line 19 by the D-CPU, which may interrogate the state and perform any other necessary function. It may read (or write) into the memory of the H-computer by saving the state of the H-CPU, sending a new state to cause a word of memory to enter a general register (or transfer the contents of a register into memory) and then restoring the internal state of the H-CPU. All of
this is accomplished via the LSSD register (41) in the H-computer.

Figure 3 shows the interface unit 13 consisting of three sections, i.e., start-stop control 31, LSSD control 33, and shift control 35.

Start-stop control unit 31 receives signals which control the mode of operation of the H-computer and a signal for controlling starting of the H-CPU. It receives input from the H-CPU which determines when the H-CPU is at an "end of cycle" or when it is at an "end of an instruction". The start-stop control unit 31 issues the "run H-CPU" signal which controls whether the H-CPU is to run or is to be stopped. The logic of start-stop control unit 31 also issues the interrupt to the D-CPU notifying it that the host has halted at which time the D-CPU may interrogate the state of the H-CPU.

LSSD control unit 33 operates under three commands: "receive state", "send state" and "send mask" transferred over lines 36, 37 and 38 respectively. These commands allow the D-CPU to receive state data under the control mask, send state data under the control mask or set a new mask in the interface. LSSD control unit 33 uses the data bus to obtain state information and masks from the D-computer. It uses the strobe and enable signals from the DMA in the D-computer as conditions for reading or writing the data bus. The shift control unit 35 contains a counter 75 which issues a signal every time that the LSSD register is to be shifted. After 32 shifts it issues a signal "S32" (line 76) allowing a word to be written into or read from the D-computer under control of the mask. When all state information has been transferred, a "transfer complete" signal is issued on line 29 to the D-computer. The "receive state", "send state" and "send mask" signals are used to specify the read memory and write memory commands to the DMA.

LSSD control 33 receives a stream of bits from the LSSD register 41 in the H-CPU and issues a stream of bits to the LSSD register. The "receive state" command determines that a preselected number of words (32 bits) from the stream coming from the LSSD register will be transferred to the D-computer memory 24. The "send state" command specifies that a set of state words from the memory of the D-CPU will be merged with the current state of the H-CPU and sent to the LSSD register in the H-CPU.

Figure 4 illustrates how the H-CPU signals the interface when it is at the end of a cycle or at the end of an instruction. The interface responds with a "run H-CPU" signal causing the H-CPU to resume processing or stay in a halt state. The two LSSD data lines from the interface are used to obtain a bit from the LSSD register 41 and at the same time send a bit into the LSSD register 41 under the control of the shift signal. The LSSD (Level Sensitive Scan Design) register 41 is the means of shifting latch data into and out of the H-CPU. The latches hold the internal state of the CPU. For still further description of the LSSD design concept, see E. B. Eichelberger and T. W. Williams, "A Logic Design Structure for LSI Testability", 14th Design Automation Conference, New Orleans, 1977, p. 482—488.

As stated previously, the present system architecture for a diagnostic/debug arrangement is concerned with the overall architecture necessary for interfacing with the host CPU and obtaining the requisite state data from the LSSD registers and placing either the same or modified state data back into the LSSD registers. It will also be noted that the D-computer itself may be a complex standalone computer concerned with performing sophisticated diagnostic routines in its own right. The significance of the architecture of the present invention is that a means is provided for transferring requisite state data between the host computer 12 and D-computer 11 in an extremely efficient and self-contained manner. It is the function of the command control unit 22 of Figure 5, to detect when the D-computer wishes to take some action involving the state data located in or to be stored in the LSSD register 41 of the host computer. This is done by means of special address patterns located in the address portion of the D-computer instruction. For the present embodiment, as will be explained subsequently, the particular bit pattern appearing in a first portion of the address field indicates to command control unit 22 that an instruction involving unit 22 has been detected. The particular instruction or command is detected by analyzing a second portion of the address field. Thus as instructions are being continually processed by the D-computer and the D-CPU in particular, the instructions continuously appear on the address portion of the bus 64 entering command control unit 22. However, of all the many addresses which are produced by the D-CPU and placed on the data address R/W bus, only the particular bit pattern which indicates that a command is to be given to the present data acquisition architecture will cause command control unit 22 to be activated as will be well understood.

Figures 5 and 6 show the details of the command control structure in the D-computer. When the D-CPU issues an instruction such as store general register N (in the D-computer) in location A in memory, where A has the particular format (14 high order one bits) described below, then the signal on one of the seven command lines is raised.

Figure 5 shows that the address bus 64 is split from the data address R/W bus and is then split into the high order and the low order. The high order part is fed into a 14-input AND circuit 62 and if the output is true, the signal is used to set the operation register 65 with the lower order part of the address bus. The operation register output is sent via the commands
lines 15 to the interface. The “receive state” and the “send state” lines are ORed in OR 60 and then ANDed in AND 61 with the output of the 14-input AND circuit 62. This signal is in turn used to reset latch 63 which determines whether the D-CPU is to run or is to remain halted. The latch 63 is set by the “transfer complete” signal from the interface.

Figure 6 shows the address of such an instruction. When the instruction is executed, the address part is placed on the address bus. The address consists of a high order part of 14 bits and a low order part of 7 bits. The high order part of the address is all ones. This signals to command control unit 22 that the D-CPU is issuing one of seven commands, which of the seven commands is coded in the low order 7 bits of the address. Only one of these bits is on to identify the indicated command.

Figure 7 shows the details of shift control unit 35. The main component is a counter 75 which is started via OR circuit 71 by either the “receive state” or “send state” command. It will be noted that the counter contains a system clock which is chosen to have a frequency compatible with the system clock of the host machine so that its shifting rate will be adaptable for controlling the LSSD registers contained therein. The counter is of any type well known in the art and is shown having three outputs. These are labelled as “1”, “32” and “1024”. The output labelled “1” is the basic shift pulse utilized to effect the shifting of bits within the LSSD control unit 33 in the interface unit and also the LSSD registers 41 within the H-computer. This line would be the simple output of the basic clock. The output labelled “32” produces the S32 pulse on line 76. This line produces a pulse after every 32 shift pulses (operates mod. 32). As will be understood in referring to the LSSD control unit 33 of Figure 9, the S32 pulse controls the gating of 32 bits of data parallel into or out of the input and output registers 96, 98 and controls the shifting of the mask register (97).

Finally, the output from the counter labelled “1024” indicates that a complete state data transfer has occurred between the LSSD register 41 in the H-CPU and the interface unit. Thus, this output on line 29 is used to signal to the D-computer that a data transfer is complete and also resets the counter. Output “1” (low order bit) of the counter gives a shift signal every time the bit is to be shifted into LSSD register 41. As stated previously, output “32” is the S32 signal, and is issued whenever a word (32 bits) may be transmitted to or from the D-computer. The LSSD register 41 is assumed to contain 32 times 32 bits or 1024 total bits and thus output “1024” of the counter signals the “end of the transfer” of state data to or from the LSSD register (if the H-computer requires more than 1024 bits the extension can be made in a straightforward manner). This “transfer complete” signal on line 29 resets the counter and is sent to the D-computer to resume the running of the D-CPU. The “read memory” signal is generated when the AND circuit 72 receives the “send state” command and the S32 signal or when the “send mask” signal is present (via OR circuit 73). The “write memory” signal is generated by AND circuit 74 when a “receive state” command coincides with the appropriate time signal from the counter which is the S32 signal.

Figure 8 shows the start-stop control unit 31. The three lines representing the mode commands (“set cycle end mode”, “set instruction end mode” and “set continuous mode”) are used to control the three latches L1, L2 and L3. Only one of the latches is on at a time. Each mode command turns on one of these latches and resets the other two latches via OR gates 84, 85 and 86.Latch L4 determines whether an “interrupt” signal is to be sent to the D-computer on line 19, and the status of the “run H-CPU” signal. It is set via OR circuit 80 by the “start H-CPU” signal or the “continuous mode” signal from mode latch L3. The condition for resetting latch L4 is the logical OR of two conditions via OR circuit 81. In the first sub-condition the H-CPU is at an “end of cycle” and the cycle mode latch L1 is on. The second sub-condition is that the H-CPU is at an “end of instruction” and the instruction mode latch L2 is on. The two aforementioned subconditions are implemented as inputs to OR circuit 81 via an output from one of the AND circuits 82 and 83 respectively.

Figure 9 shows LSSD control unit 33. A stream of bits is received on “LSSD out” and passed with possible modification to “LSSD in”. Under the control of bit 32 (high order bit) of the mask register 97 a stream of bits from the output register 98 may be substituted. The bit stream from “LSSD out”, being loaded into the input register 96 may be transferred to the D-computer under control of mask register 97 (bit 32).

LSSD control unit 33 is under control of the three commands; “receive state”, “send state” and “send mask” received on lines 36, 37 and 38, respectively. During the “receive state” operation, the bit stream from LSSD out enters input register 96 under the control of the “shift” signal. If bit 32 of mask register 97 is a “1” and three other conditions are in the “1” state, AND circuit 90 is enabled and its output enables gate 33 whereby the contents of input register 96 is placed in parallel on the data bus 25. The three other conditions (inputs to AND 90) for gating the contents of the input register to the data bus are: S32 on line 76 is up (32 bits have been received), the “receive state” command is in operation and finally, that the “enable” signal on line 27 has been received from the DMA.

If the “send state” command is in operation, bit 32 of the mask register determines whether bits from the output register are to be placed on the “LSSD in” lines. Bit 32 of the mask register
and its complement are fed to AND circuits 94 and 95 for gating. A bit stream will pass through one of these gates. In one case the stream comes from “LSSD Out” and in the other case the stream comes from the output register which is under control of the shift signal. Setting or loading the output register is determined by three conditions: the “send state” command is in effect, S32 time has arrived and the DMA “strobe” signal on line 26 is present indicating the data bus is available. These three signals (together with a “1” in bit 32 of the mask register) enable AND circuit 92.

The “send mask” command loads a mask from the D-computer into mask register 97. The mask register is a circular shift register under control of the S32 signal. The new mask is obtained from the D-computer when the “send mask” command is performed and the “strobe” signal has been received from the DMA indicating that the data bus is available for transferring the mask to the mask register.

For the “send state” and “receive state” commands, 32 shifts bring a word of 32 bits from the LSSD register 41. Bit 32 of the mask register 97 determines whether a word of state data is to be transferred to or from the D-computer. S32 rotates the mask register contents and the process repeats for the next 32 bits from the LSSD register 41 until 1024 bits have been transferred to and from the LSSD register.

A typical set of instructions which the D-CPU would cause to be placed on the Data Address RW Bus to be picked up by the command control unit 22 would be, for example, that shown in Table 1. The table shows the function required of the system, it being understood that the instruction format illustrated in Figure 6 would be followed. In all of these instructions, the high order 14 bits would be all “1” and the particular command or operation desired would be specified by setting one of the lower order 7 bits of the address field to a “1”. The table illustrates what would happen in each of the three basic units as shown in Figure 1 (the H-computer, the interface unit, and the D-computer) for each separate instruction. Thus, the example places one of three operating mode commands (end of cycle) in the start/stop control unit 31, loads a mask into mask register 97, causes data to be transferred from LSSD register 41 into the interface unit 13 and subsequently causes different state data to be returned to the LSSD register 41 in the H-computer 12.

<table>
<thead>
<tr>
<th>Diagnostic computer</th>
<th>Interface</th>
<th>Host computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>5. Send State</td>
<td>6. Reads State from D-Computer and Shifts Output Register to LSSD</td>
<td>11. Shifts LSSD from Host and Writes Into D Memory</td>
</tr>
<tr>
<td>7. Start Host CPU</td>
<td>10. Receive State (From LSSD Register Under Same Mask)</td>
<td>14. Transfers State to LSSD</td>
</tr>
<tr>
<td>12. Analyze State</td>
<td>13. If Error Exists Then (Enter a Diagnostic Routine, Save State, Send New Diagnostic State) Else (Continue)</td>
<td>16. Runs One Cycle and Signals “End of Cycle”.</td>
</tr>
<tr>
<td>15. Start Host CPU</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In the table it should be understood that only those items appearing under the D-computer column are actual instructions interpreted by the command control unit as described previously. Each of the numbered occurrences 1 through 16 is indicated as occurring sequen-

tially since the time axis is considered to be the vertical direction in the table. However, it will be appreciated that many of the items occur substantially simultaneously. Thus, item 1, which is an instruction to “Set Cycle End Mode” causes the interface unit to set the appropriate logic in
the start/stop control unit of the interface unit (item 2). Similarly instruction 3 "Send Mask" causes item 4 in the interface unit to occur which is the loading of the actual mask from the D-computer memory into the mask register in the interface.

Item 5 "Send State" causes the desired state configuration to be read from the D-computer memory into the output register and then causes the contents of the output register to be shifted into the LSSD registers of the H-computer (item 6). Item 7 is again an instruction in the D-computer which signals that the host CPU should resume operation. This causes item 8 to occur whereby the host runs for one cycle and signals on its "Cycle End" output line that an end of cycle has been reached.

Item 9 of the interface unit causes an "interrupt" signal to be sent to the D-CPU. This is actually the end of one cycle sequence and at this time the next instruction, 10, which would be sent by the D-computer controls would be a "receive state" instruction which means that the state data in the LSSD registers should be transferred to the LSSD control unit (assumedly under the same mask specified in item 3).

Item 11 in the interface unit indicates that the LSSD contents are entered into the LSSD control unit (input register) under the mask and subsequently written into the D-computer memory for analysis. Instruction item 12 in the D-computer causes the state data to be analyzed. Instruction 13 is essentially a diagnostic analysis routine forming no part of the present invention, however, as will be appreciated, if an error is detected then some sort of a diagnostic routine would be entered which would cause the erroneous state to be saved in the D-computer and a new diagnostic state returned to the H-computer (at some point in time). It is possible that, depending upon the nature of the error, the H-computer would remain interrupted until the difficulty is resolved.

Assuming that new state data is to be transferred to the LSSD registers as a result of the diagnostic routine of item 13, then specified data is read from the D-computer memory and shifted into the LSSD registers of the host (item 14). Instruction item 15 causes the host CPU to be restarted, as in instruction 7 above and in item 16 the host runs one cycle and signals "end of cycle" as in item 8 above.

It will be apparent from the above description of the disclosed diagnostic/debug arrangement that many changes could be made in the form and details of the data acquisition hardware architecture of the present system without departing from the basic concepts present. Similarly, additional enhancements could be made to the present system for achieving greater economy of operation neither within the host computer being monitored, or within the diagnostic computer itself.

One such enhancement which increases the performance of the H-computer and the efficiency of the operation of the D-computer follows.

If the H-computer is under diagnosis as in Figure 1, the readout of the machine state (LSSD registers) occurs at every cycle or every instruction execution as described. The diagnostic computer is invoked at every read-out time, resulting in the execution of a number of instructions in the D-computer. This results in a degradation of the performance in the H-computer of, perhaps, several orders of magnitude, depending upon the relative speeds of the two computers. If the D-computer must indeed examine every readout, then that degradation is the inherent cost of this method of diagnosis. It is, however, likely that the D-computer is not interested in every readout. It may, for example, be interested only in the operation "floating-point multiply" or in any operation which has, as its hex digit, the character "F" or any of a number of other probably low frequency items.

Thus, it would be desirable to so design the controls of the present diagnostic/debug arrangement so that only those operations in which the system is currently interested, would be examined and, of those, only anomalous situations would cause the invoking of the D-computer. This could be done by utilizing some sort of early selection stage wherein the readout from the H-computer might be placed in a first register such as the state data input register 101. The diagram in Figure 10 illustrates the technique. Another register 103 containing an "Early Selection Mask" would be provided and the contents of the state data input register 101 and the early selection mask register 103 could be bitwise ANDed resulting in a bit string of zeros everywhere except for those fields isolated by the Early Selection Mask. This resultant bit string would then be compared with the contents of a third register 105 (a value register or collection of value registers organized as an associative store) which contains a bit configuration in which the D-computer is interested. As a result of a comparison, (i.e., an equal condition), the D-computer would be invoked, and control would proceed as with the present system so that the complete state data contents could be transferred into the D-computer. If, however, the comparison results in an unequal condition, the D-computer is not invoked and the H-computer is immediately released. It is presumed that the D-computer would load early selection mask register 103 and value register 105 with predetermined bit strings which will require its invocation only when there are occurrences of state changes which it wishes to examine, and if these occurrences are infrequent, the performance degradation in the H-computer will be correspondingly less severe.

The use of such an Early Selection enhancement could be made modal, so that when the diagnostic system is not in the early selection
mode, the system would run as described previously. Stated differently, if such an Early Selection enhancement were used, the circuitry could be utilized solely for the purpose of indicating to the system that a desired anomalous condition had occurred and at that point the previously described data transfer of the state data into the registers, 32 bits at a time, could occur under a special interrupt instruction sent to the D-computer by the Early Selection hardware.

It will of course also be appreciated that the above described registers could themselves be modified for direct transfer of data, into and out of the D-computer.

Industrial applicability

The herein disclosed diagnostic/debug arrangement has application for the monitoring of any host computer utilizing the LSSD design concept. When the D-computer is used for diagnostics it may be used to exhaustively test the hardware portion of the host computer. It may be used to find the specific hardware faults causing the detected error conditions. If the H-computer is a high performance machine with interruptible instructions, the end of cycle mode allows a view of the state of the machine during interrupt.

As a debug arrangement, it acts as a probe into the host CPU and its memory without interfering with the environment of the H-CPU. It could easily determine if a given instruction is being executed or if a particular branch is being taken. A dual compiler could be written for the H-CPU and the D-CPU. Such a compiler would allow symbolic names which are commonly used for the two programs to be resolved to the same locations in the H-computer memory.

The present diagnostic/debug arrangement thus has application whenever it is desired to diagnose a host computer or debug a program. It being noted that the system can be attached to the host at any time whether the host is new and being brought up for the first time, or has been in service for many years.

When the host computer works properly, the interface unit may be removed and a simple unit replaces it so that the host can run normally. The interface and the diagnostic computer can then be used to service other defective host computers. They need not be permanently connected to a particular host computer.

Claims

1. A diagnostic and debugging arrangement for a host computer (12) incorporating a level-sensitive scan register (41), and having interface lines (16, 17) for transfer of level-sensitive scan data and for sense and control signals, said diagnostic and debugging arrangement comprising a diagnostic computer (11) and an interface unit (13) interposed between said host computer and said diagnostic computer in data exchange relationship with both; wherein said interface unit (13) includes register means (96, 98) for receiving state data from the level-sensitive scan register (41) of the host computer to be transferred to said diagnostic computer, and for storing state data from the diagnostic computer to be stored in predetermined locations in the level-sensitive scan register (41) of said host computer, and control means (33, 35) actuable by said diagnostic computer to cause state data to be selectively transferred between said host computer and said diagnostic computer via said register means (96, 98); and wherein said diagnostic computer (11) includes control means (21, 22, 23) for effecting the execution of predetermined diagnostic programs which require the access of state data from and the transfer of state data to the host computer, characterized in that

— said control means (33, 35) in the interface unit (13) executes independently of the host computer and the diagnostic computer the shifting and selection operations necessary for the selective transfer of state data, upon receipt of respective command signals from the diagnostic computer (11), and that
— said control means (21, 22, 23) in the diagnostic computer comprises means (62) connected to the address bus (64) of the diagnostic computer for interpreting certain address data of its instructions as specific commands, and means (15, 65) for furnishing respective command signals which determine the conditions and direction of the state data transfer, to said control means (33, 35) in the interface unit,

2. An arrangement in accordance with claim 1, characterized in that

— early selection control means (Figure 10) are provided in the interface unit (13) for specifying certain state data conditions which must exist before the diagnostic computer (11) will be actuated, said early selection control means including
— state data input register means (101) in which state data from the level-sensitive scan register (41) in said host computer (12) is temporarily stored,
— early select mask register means (103) loadable from said diagnostic computer (11) for specifying those portions of the state data in which the diagnostic computer is interested,
— value register means (105) for specifying data conditions in those portions of the state data previously specified by the mask register contents which must exist prior to causing actuation of said diagnostic computer, and
— logic evaluation means for determining from the contents of these register means, independently from the host computer and the
diagnostic computer, whether said certain state data conditions exist, and for issuing a respective indication signal.

3. An arrangement in accordance with claim 1, characterized in that

— said interface unit (13) further includes a mask register (97) for storing a control mask which determines which segments of the state data being exchanged between the host computer (12) and the diagnostic and debugging arrangement are to be transferred to the diagnostic computer (11) before being returned to the host computer, and

— said control means (21, 22, 23) in said diagnostic computer includes means (25, 27, 38) for loading a predetermined mask into said mask register (97) to effect said controlled transfer of data between said host computer and said diagnostic computer.

4. An arrangement in accordance with claim 3, for a host computer (12) wherein the entire state data within the level-sensitive scan register (41) have a total length equal to m data words each containing n bits, characterized in that

— said mask register (97) in said interface unit (13) contains m bit positions, and

— control means (35, 76, 90, 92, 94, 95) are provided in said interface unit for accessing consecutive bit positions of said mask during the transfer of data between said host computer and said diagnostic computer, and for using the contents of each single bit position of said mask register to control the transfer of n bits of state data.

Patentansprüche

1. Diagnose- und Fehlerbeseitigungs-Anordnung für einen Haupt-Rechner (12), welcher ein Testschiebebiten-Register (level-sensitive scan register) (41) enthält und welcher Schnittstellenleitungen (16, 17) aufweist für die Übertragung von Testschiebebiten-Register-Daten (level-sensitive scan data) sowie von Abtast- und Steuersignalen, wobei die Diagnose- und Fehlerbeseitigungs-Anordnung einen Diagnose-Rechner (11) enthält und eine Schnittstelleinheit (13), welche zwischen dem Haupt-Rechner und dem Diagnose-Rechner angeordnet ist und mit beiden in Daten austausch-Verbindung steht; wobei die Schnittstelleinheit (13) Register (96, 98) enthält zum Empfang von Zustandsdaten vom Testschiebebiten-Register (41) des Haupt-Rechners, welche zum Diagnose-Rechner zu übertragen sind, und zum Speichern von Zustandsdaten vom Diagnose-Rechner, welche in vorbe-stimmten Plätzen des Testschiebebiten-Registers (41) des Haupt-Rechners zu speichern sind, sowie Steuereinrichtungen (33, 35), welche vom Diagnose-Rechner aktivierbar sind, um eine selektive Uebertragung von Zustandsdaten zwischen dem Haupt-Rechner und dem Diagnose-Rechner über die Register (96, 98) zu bewirken; und wobei der Diagnose-Rechner (11) Steuereinrichtungen (21, 22, 23) aufweist, um die Ausführung vorbestimmter Diagnoseprogramme zu bewirken, für welche der Zugriff auf Zustandsdaten aus dem und die Uebertragung von Zustandsdaten in den Haupt-Rechner erforderlich ist, dadurch gekennzeichnet,

— das die Steuereinrichtungen (33, 35) in der Schnittstelleinheit (13) die Verschiebe- und Auswahloperationen, welche für die selektive Uebertragung von Zustandsdaten erforderlich sind, unabhängig vom Haupt-Rechner und vom Diagnose-Rechner (11) ausführen, und

— dass die Steuereinrichtungen (21, 22, 23) im Diagnose-Rechner eine mit dem Adressbus (64) des Diagnose-Rechners verbundene Vorrichtung (62) enthalten, um bestimmte Adressdaten seiner Befehle als besondere Kommandos zu interpretieren, sowie Vorrichtungen (15, 65) zur Abgabe entsprechender Kommandosignale, welche die Bedingungen und die Richtung der Zustandsdatenübertragung festlegen, an die Steuereinrichtung (33, 35) in der Schnittstelleinheit.

2. Anordnung gemäss Anspruch 1, dadurch gekennzeichnet,

— dass in der Schnittstelleinheit (13) eine Früh-Auswahlvorrichtung (Fig. 10) vorgesehen ist, um bestimmte Zustandsdaten-Bedingungen zu bestimmen, welche vorliegen müssen, bevor der Diagnose-Rechner (11) aktiviert wird, wobei diese Früh-Auswahlvorrichtung enthält:

— ein Zustandsdaten-Eingaberegister (101), in welchem Zustandsdaten vom Testschiebebiten-Register (41) im Haupt-Rechner (12) vorübergehend gespeichert werden,

— ein Früh-Auswahl-Maskenregister (103), welches vom Diagnose-Rechner (11) ladbar ist, um die Teile der Zustandsdaten anzugeben, an welchen der Diagnose-Rechner interessiert ist,

— ein Werteregister (105) zur Angabe von Datenbedingungen in denjenigen Teilen der Zustandsdaten, die vorher vom Inhalt des Maskenregisters bestimmt wurden, welche vorliegen müssen, bevor der Diagnose-Rechner aktiviert wird, und

— eine logische Auswerteeinrichtung, um aus dem Inhalt der Register, unabhängig vom Haupt-Rechner und vom Diagnose-Rechner, zu bestimmen, ob die betreffenden Zustandsdatenbedingungen vorliegen, und um
ein entsprechendes Anzeigesignal abzugeben.

3. Anordnung gemäß Anspruch 1, dadurch gekennzeichnet,

— das die Schnittstelleneinheit (13) ferner ein Maskenregister (97) enthält zur Speicherung einer Steuermaske, die festlegt, welche Segmente der Steuordaten, die zwischen dem Haupt-Rechner (12) und der Diagnose- und Fehlerbeseitigungsanordnung ausgetauscht werden, an den Diagnose-Rechner (11) zu übertragen sind, bevor sie an den Haupt-Rechner zurückgegeben werden, und

— dass die Steuereinrichtungen (21, 22, 23) im Diagnose-Rechner Vorrichtungen (25, 27, 38) enthalten, um eine vorbestimmte Maske in das Maskenregister (97) zu laden, um so die gesteuerte Uebertragung von Daten zwischen dem Haupt-Rechner und dem Diagnose-Rechner zu bewirken.

4. Anordnung gemäß Anspruch 3, für einen Haupt-Rechner (12), bei dem die gesamten Zustandsdaten im Testschleifeketten-Register (41) eine gesamte Länge von n Datenwortern zu je n Bits haben, dadurch gekennzeichnet,

— dass das Maskenregister (97) in der Schnittstelleneinheit (13) im Positionen umfasst, und

— das Steuereinrichtungen (35, 76, 90, 92, 94, 95) in der Schnittstelleneinheit vorgesehen sind, um zu aufeinanderfolgenden Bitpositionen der Maske während der Uebertragung von Daten zwischen dem Haupt-Rechner und dem Diagnose-Rechner zuzugreifen, und um den Inhalt jeder einzelnen Bitposition des Maskenregisters zur Steuerung der Uebertragung von n Bits der Zustandsdaten zu verwenden.

Revendications

1. Un dispositif de diagnostic et de suppression d’erreur pour un ordinateur principal 12, comportant un registre de balayage sensible au niveau 41 et des lignes d’interface 16, 17 pour le transfert des données de balayage sensibles au niveau et pour les signaux de détection de commande, ledit dispositif de diagnostic et de suppression d’erreurs comprenant un ordinateur de diagnostic 11 et un bloc d’interface 13 interposés entre ledit ordinateur principal et ledit ordinateur de diagnostic de façon à permettre un échange mutuel des données, dans lequel ledit bloc d’interface 13 comprend un moyen de registre 96, 98 pour recevoir les données d’état depuis le registre de balayage sensible au niveau 41 de l’ordinateur principal à transférer audit ordinateur de diagnostic, et pour mémoriser les données d’état provenant de l’ordinateur de diagnostic et à stocker dans des emplacements prédéterminés du registre de balayage sensible au niveau 41 dudit ordinateur principal, et un moyen de commande 33, 35 pouvant être excité par ledit ordinateur de diagnostic afin de provoquer le transfert sélectif des données d’état entre ledit ordinateur principal et ledit ordinateur de diagnostic à travers ledit moyen de registre 96, 98; et dans lequel ledit ordinateur de diagnostic 11 comprend des moyens de commande 21, 22, 23 pour procéder à l’exécution des programmes de diagnostic prédéterminés qui exigent l’accès des données d’état depuis l’ordinateur principal et le transfert des données d’état vers ce même ordinateur, caractérisé en ce que:

— ledit moyen de commande 33, 35 du bloc d’interface 13 exécute indépendamment de l’ordinateur principal et de l’ordinateur de diagnostic les opérations de décalage et de sélection nécessaires pour le transfert sélectif des données d’état à réception des signaux d’instruction respectifs provenant de l’ordinateur de diagnostic 11, et en ce que

— ledits moyens de commande 21, 22, 23 de l’ordinateur de diagnostic comportent un moyen 62 connecté au bus d’adresse 64 de l’ordinateur de diagnostic, pour interpréter certaines données d’adresse de ces instructions, sous forme d’instructions spécifiques et un moyen 15, 65 pour fournir des signaux d’instruction respectifs qui déterminent les conditions et le sens de transfert des données d’état vers ledits moyens de commande 33, 35 du bloc d’interface.

2. Un dispositif selon la revendication 1, caractérisé en ce que des moyens de commande de présélection (Fig. 10) sont disposés dans le bloc d’interface 13 pour spécifier certaines conditions de données d’état qui doivent exister avant que l’ordinateur de diagnostic 11 soit mis en route, ledits moyens de commande de présélection comprenant:

— un moyen de registre d’entrée de données d’état 101 dans lequel les données d’état proviennent du registre de balayage sensible au niveau 41 dans ledit ordinateur principal 12 sont temporairement stockées;

— un moyen de registre de masque de présélection 103 pouvant être chargé depuis ledit ordinateur de diagnostic 11 pour spécifier les portions des données d’état qui concernent l’ordinateur de diagnostic;

— un moyen de registre de valeur 105 pour spécifier les conditions des données dans les portions des données d’état précédemment spécifiées par le contenu du registre de masque, et qui doivent exister avant de provoquer la mise en œuvre dudit ordinateur de diagnostic et

— un moyen d’évaluation logique pour déterminer à partir du contenu de ces moyens de registre, et indépendamment de l’ordinateur
principal et de l’ordinateur de diagnostic, s’il existe certaines conditions des données d’état, et pour émettre un signal indicatif respectif.

3. Un dispositif selon la revendication 1, caractérisé en ce que:
— ledit bloc d’interface 13 comporte en outre un registre de masque 97 pour mémoriser un masque de commande qui détermine quels seront les segments de données d’état échangés entre l’ordinateur principal 12 et le dispositif de diagnostic et de suppression d’erreurs qui doivent être transférés à l’ordinateur de diagnostic 11 avant d’être renvoyés à l’ordinateur principal, et
— lesdits moyens de commande 21, 22, 23 dudit ordinateur de diagnostic comprennent des moyens 27, 38, pour charger un masque prédéterminé dans ledit registre de masque 97 afin d’effectuer le transfert contrôlé des données entre ledit ordinateur principal et ledit ordinateur de diagnostic.

4. Un dispositif selon la revendication 3, pour un ordinateur principal 12 dans lequel les données d’état tout entières se trouvant dans le registre de balayage sensible au niveau 41 ont une longueur totale égale à m mots de données dont chacun contient n bits, caractérisé en ce que:
— ledit registre de masque 97 dudit bloc d’interface 13 contient m positions de bits et
— des moyens de commande 35, 76, 86, 92, 95, 95 sont disposés dans ledit bloc d’interface pour pouvoir accéder aux positions consécutives de bits dudit masque pendant le transfert des données entre ledit ordinateur principal et ledit ordinateur de diagnostic, et pour utiliser le contenu de chaque position individuelle de bit dudit registre de masque pour commander le transfert de n bits de données d’état.
FIG. 6

<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>R/W</th>
</tr>
</thead>
<tbody>
<tr>
<td>HIGH 14 BITS</td>
<td>LOW 7 BITS</td>
</tr>
<tr>
<td>VALUE I I I I . . . 1</td>
<td>VALUE ONLY 1 BIT ON</td>
</tr>
</tbody>
</table>

FIG. 5

COMMAND CONTROL

DATA ADDRESS
R/W BUS
ADDRESS
HIGH
LOW
ADDRESS
OPERATION REG.
SET
COMMANDS
RECEIVE STATE
SEND STATE
TRANSFER COMPLETE

S
L
R
28
62
A
A
61
63
CPU

22
65
7
14

29
15
7
4

4