PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification ^{4} :
G06F 3/00, 5/00, 13/18

A1

(11) International Publication Number: WO 88/ 08564

(43) International Publication Date: 3 November 1988 (03.11.88)

(21) International Application Number: PCT/US88/01166

(22) International Filing Date: 12 April 1988 (12.04.88)

(31) Priority Application Number: 041,533

(32) Priority Date: 23 April 1987 (23.04.87)

(33) Priority Country: US

(71) Applicant: COMMODORE-AMIGA, INC. [US/US]; 1200 Wilson Drive, West Chester, PA 19380 (US).

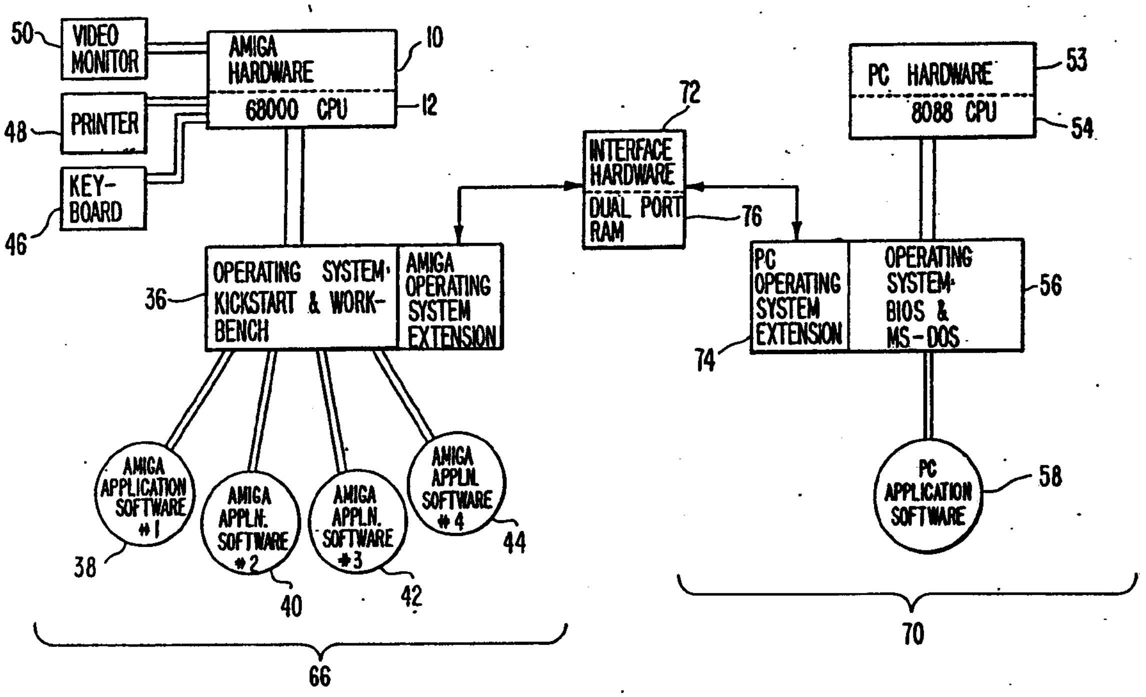
(72) Inventors: PREISS, Dieter ; Okestr. 2A, D-3175 Leiferde (DE). ULLMANN, Frank-Thomas ; Huttenstr. 9, D-3300 Braunschweig (DE).

(74) Agents: GARRETT, Arthur, S. et al.; Finnegan, Henderson, Farabow, Garrett & Dunner, 1775 K Street, N.W., Washington, DC 20006 (US).

(81) Designated States: AT, AT (European patent), AU, BB, BE (European patent), BG, BJ (OAPI patent), BR, CF (OAPI patent), CG (OAPI patent), CH, CH (European patent), CM (OAPI patent), DE, DE (European patent), DK, FI, FR (European patent), GA (OAPI patent), GB, GB (European patent), HU, IT (European patent), JP, KP, KR, LK, LU, LU (European patent), MC, MG, ML (OAPI patent), MR (OAPI patent), MW, NL, NL (European patent), NO, RO, SD, SE, SE (European patent), SN (OAPI patent), SU, TD (OAPI patent), TG (OAPI patent).

Published
With international search report.

(54) Title: A METHOD OF COMMUNICATING DATA BETWEEN THE CPU OF A HOST COMPUTER SYSTEM AND THE CPU OF A CO-PROCESSOR COMPUTER SYSTEM



(57) Abstract

A method of communicating data between the central processing unit ("CPU") (12) of a host computer system (66) and the CPU (54) of a co-processor computer system (70) is provided for allowing noncompatible application software to be used by the host computer system. The method includes using a dual port random access memory (76) shared between the CPUs (12, 54) of the two computer systems for interprocessor communication.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AT	Austria
AU	Australia
BB	Barbados
BE	Belgium
BG	Bulgaria
BJ	Benin
BR	Brazil
CF	Central African Republic
CG	Congo
CH	Switzerland
CM	Cameroon
DE	Germany, Federal Republic of
DK	Denmark
FI	Finland
FR	France
GA	Gabon
GB	United Kingdom
HU	Hungary
IT	Italy
JP	Japan
KP	Democratic People's Republic of Korea
KR	Republic of Korea
LI	Liechtenstein
LK	Sri Lanka
LU	Luxembourg
MC	Monaco
MG	Madagascar
ML	Mali
MR	Mauritania
MW	Malawi
NL	Netherlands
NO	Norway
RO	Romania
SD	Sudan
SE	Sweden
SN	Senegal
SU	Soviet Union
TD	Chad
TG	Togo
US	United States of America

A METHOD OF COMMUNICATING DATA BETWEEN THE CPU OF A HOST COMPUTER SYSTEM AND THE CPU OF A CO-PROCESSOR COMPUTER SYSTEM

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention relates, in general, to microprocessor controlled video games and personal computers and, more particularly, to the communication between the central processing unit ("CPU") of a host computer system and that of a co-processor computer system to provide application software compatibility.

Description of Related Art

A large number of personal computers are available on the market today. Typically, each of the personal computers is restricted in its operation to application software designed specifically for the particular computer system. Also typically, the library of available application software for one particular computer system does not contain all of the programs found in the libraries of application software of other computer systems. When application software is not available for a specific computer system, it would be most beneficial to be able to utilize the different but noncompatible application of another computer system.

Furthermore, a specific computer system may have superlative processing speed, enhanced graphics, enhanced sound, and/or other high performance capabilities not available in the computer system which enjoys a large or different library of application software. In this situation, it would be useful if the noncompatible application software could be used within the computer system having the high performance capabilities so that the better processing enhancements provided by the high performance capabilities could be used with the noncompatible software.

A specific example is illustrated by the Amiga computer system design, manufactured, and marketed by Commodore Business Machines, Inc. of West Chester, Pennsylvania. The Amiga system is a low-cost, high performance computer with advanced graphics features, sound features, and high speed performance. However, the Amiga system is not IBM-PC/XT ("PC") compatible and, thus, cannot use the library of PC application software which is readily available on the market.

As will become evident below, the disclosure focuses on the Amiga computer and a method of making it PC compatible. The invention disclosed and claimed, however, is not limited to an Amiga/PC environment. That environment is merely presented as a specific, preferred one for purposes of explanation.

The Amiga personal computer has a low-cost, high performance graphics and sound system for state of the art video game and personal computer applications. The system includes three custom integrated circuits controlled by a Motorola 68000 16-32/-bit microprocessor as the CPU. These custom chips, designated Agnus, Denise, and Paula, provide extraordinary color graphics on a standard TV or on a color video minotor with arcade quality resolution and depth to display video games, cartoons, low resolution photographs and up to 80 characters of text on the screen. Additionally, the sound circuits can duplicate complex wave forms on each of four channels, matching commercial synthesizers in quality. Furthermore, the Amiga computer system has multitasking capability, that is, it can perform more than one program at a time and is interrupt driven.

A detailed description of the hardware and operation of the Amiga computer system is found in the following copending application which are hereby incorporated by reference:

1. "Video Game and Personal Computer", Serial No. 756,910, filed July 19, 1985;

2. "Display Generator Circuitry for Personal Computer System", Serial No. 886,796, filed July 18, 1986;

3. "Peripheral Control Circuit for Personal Computer", Serial No. 886,614, filed July 18, 1986; and

4. "Data Input Circuit With Digital Phase Locked Loop", Serial No. 886,615, filed July 18, 1986.

Presently, the Amiga computer system is available as the Amiga 1000 computer and the Amiga 2000 computer. For the purposes of this disclosure, the differences between the two

Amiga computer systems are not relevant, and reference will be made to the Amiga 2000 computer system for simplicity. The hardware of an Amiga 2000 is shown in block diagrammatic form in Fig. 1. Fig. 2 shows in simplified block diagrammatic form the similarities and differences between the hardware of the Amiga 1000 computer system designated element 8 and the Amiga 2000 computer system designated element 10, with respect to data traffic. Specifically illustrated in Fig. 1 is the Amiga computer system's hardware and buses, collectively designated as element 10. The hardware includes the Motorola 68000 CPU 12, the three custom chips (that is, Angus 14, Denise 16, and Paula 18), and the 100 pin expansion Amiga bus slots 20. Also shown are the ports for the peripherals of computer system 10 such as keyboard port 22, parallel port 24, serial port 26, and video port 28. Additionally, an 86 pin local expansion CPU bus slot 30 and associated data, address, and control buses are illustrated.

The entire interrupt driven Amiga computer system 34 with its software is shown diagrammatically in Fig. 3. The software includes the Kickstart and Workbench operation system 36 and various available Amiga application software packages 38, 40, 42, 44 to run on the multi-tasking system 34. Also shown are keyboard 46, printer 48 and video monitor 50 connected to the appropriate ports 22, 24, and 28.

Diagrammatically presented in Fig. 4 is a PC compatible computer system 52 including PC hardware 53 with an Intel 8088 CPU 54, PC operating system 56 (that is, BIOS and MS-DOS), and a selected PC application software package 58, such as IBM's "Flight Simulator" program. Since the PC system itself is not multi-tasking, only one application program can be run on that system at one time. Like the Amiga system, the PC system is interrupt driven. Also shown are various peripherals for PC compatible system 52 such as keyboard 60, printer 62, and video monitor 64.

A review of the information provided in the copending applications and the figures discussed above provides a person of ordinary skill in the art with sufficient information on the Amiga computer system to understand the structure and operation of the system. Therefore, familiarity with that detailed information will be presumed for the purposes of the present discussion and, thus, will not be reiterated here. Likewise, familiarity with the hardware and operation of a PC computer system is presumed.

While the Amiga computer system has high performance capabilities, including superlative graphics and high processing speeds, it cannot, standing alone, run PC application software. By extending the Amiga computer system to be compatible with PC application software, not only can additional programs be run on the Amiga computer sytsem, but also the high performance processing capability of the Amiga computer system can enhance the PC application software.

From the above discussion, it is apparent that there is a great need for a method for allowing application software from a normally noncompatible computer system to be run in a host computer system.

It is, thus, an object of the invention to provide a method for allowing normally noncompatible application software to be utilized by a host computer system.

Another object of the invention is to provide a method for high speed communication of data between the central CPU of a host computer system and the CPU of a co-processor computer system which processes application software which is not compatible with the host computer system.

Still another object of the invention is to provide a method for providing data communication between a host computer system and a co-processor computer system in such a manner that the communication process is transparent to the host computer system user and to the co-processor computer system.

Still another object of the invention is to provide a method of creating a PC environment in a co-processor computer system to run PC application software for use in a host computer system which is normally not PC compatible.

Other objects and features of the invention will further become apparent with reference to the accompanying drawings and the detailed description of the invention or may be learned by practice of the invention.

SUMMARY OF THE INVENTION

To achieve the foregoing objects and in accordance with the purpose of the invention, as embodied and broadly described herein, a method of communicating data between the central processing unit (CPU) of a host computer system and the CPU of a co-processor computer system, the method using a dual port memory connected to the data and address buses of the CPUs, is provided. The method includes requesting in one of the two computer system one of a plurality of functions to be performed; generating, using the requesting computer system, parameter data representing the requested function; transferring, using the requesting computer system, the parameter data to the dual port memory; storing the parameter data in the dual port memory, issuing, using the requesting computer system, an interrupt to the other one of the two computer systems, the interrupt indicating to the other one of the two computer systems the type of function requested, the parameter data in the dual port memory based on the interrupt received to perform the requested function.

The method also includes the steps of issuing to the requesting computer system at the completion of the requested function, using the other one of the two computer systems, an acknowledgement of the completed requested function and erasing the stored data from the dual port memory using the other one of the two computer systems.

The method further includes the step of processing requests for access to the dual port memory made by the CPUs one at a time on a first-come, first-serve basis. Additionally, the method includes the step of translating the transfer data issued by one of the two computer systems into a data format usuable by the other of the two computer systems before the transfer data is stored in the dual port memory.

BRIEF DESCRIPTION OF THE DRAWINGS

The accompanying drawings, which are incorporated in and constitute a part of the specification, illustrate the method of the invention, and, together with the description, serve to explain the principals of the invention.

Of the drawings:

Fig. 1 is a block diagram of the hardware of the Amiga 2000 computer;

Fig. 2 is a block diagram illustrating the differences and similarities between the data traffic in the hardware of the Amiga 1000 and Amiga 2000 computers;

Fig. 3 is a system diagram of the Amiga 2000 computer including its software;

Fig. 4 is a system diagram of a PC compatible computer including software;

Fig. 5 is a system diagram helpful in explaining the method of the invention;

Fig. 6 is another system diagram helpful in explaining the method of the invention;

Fig. 7 is the block diagram of Fig. 2 showing the PC compatibility system in place;

Fig. 8 is a diagram showing the various portions of a dual port RAM used in a preferred embodiment of the invention; and

Fig. 9 is a block diagram of the details of the interface hardware shown in Fig. 5.

Reference will now be made in detail to the present preferred method of the invention.

DESCRIPTION OF THE PREFERRED METHOD

Referring again to the drawings, wherein like referenced characters designate like or corresponding parts throughout the several drawings, there is shown in Fig. 5 an overview of the sytem for enhancing a host computer system to allow the host system to be compatible with PC application software. In the preferred system illustrated in Fig. 5, the host computer system 66 includes the basic Amiga computer system illustrated in Fig. 3, that is system 34. The enhancement includes Amiga operating system extension 68, co-processor computer system 70, and interface hardware 72 which provides interprocessor communication between CPU 12 of the host system (Motorola 68000) and CPU 54 of the co-processor system (Intel 8088).

Co-processor computer system 70 in Fig. 5 includes hardware and software necessary to create a PC environment and also includes PC application software to run in that environment. Specifically, co-processor computer system 70 is a PC-type interrupt driven computer system with an 8088 CPU 54, RAM, ROM, and support chips less peripherals and includes a PC operating system extension 74 discussed below. Monochrome and color graphics adapters are also included in the co-processor computer system. The remainder of the co-processor system is emulated on the host system side of the interface hardware. Specifically, to run a PC application program, the Amiga system reconfigures the data being generated by input on keyboard 46 to appear as PC keyboard data for use by the PC system. The Amiga system also emulates a PC printer port on its own parallel port so that the PC system can use printer 48, and displays the PC video output in both monochrome and color on video monitor 50.

To facilitate PC application software compatibility in host computer system 66, interface hardware 72 is used between the host computer system and the co-processor computer system. The operating system extensions 68 and 74 manage and use interface hardware 72 to allow rapid interprocesor communication between CPU 12 of the host computer system 66 and CPU 54 of co-processor computer system 70. The main component of interface hardware 72 is a dual port random access memory 76 (DPRAM) connected to the data and address buses of the two CPUs 12, 54. This memory is shared by both CPUs to accomplish communication between the CPUs. Other memories, of course, could be used as long as access can be made from more than one port. Such a memory, for the purposes of this disclosure, is called a dual port memory. With the two operating system extensions 68, 74, interface hardware 72 is controlled to make the hardware appear transparent to the user of host computer system 66 and also transparent from the co-processor computer system side of the interface hardware.

In operation, the entire system appears from the two CPUs as a single co-processing system illustrated in Fig. 6. In this system, a task requested in either of the computer systems 66, 70 can, of course, communicate directly with its respective operating systems 36, 56. Additionally, however, any task requested in one operating system can communicate to the operating system of the other computer system and request that the processing system perform the requested task. For example, if a specific task is requested in the host computer's operating system 36 requesting that PC operating system 56 execute a specific function, the operating system can communicate with interface hardware 72 via operating system extension 68 and set up parameter data in a specified are of the DPRAM of the interface hardware to perform the requested function. Operating system 36 causes an interrupt to be transmitted to co-processor system 70 which interrupt indicates to the co-processor system the type of function requested. The interrupt driven co-processor computer system accepts the function request and performs the necessary program based on the interrupt received to execute the function, including accessing the parameter data in the DPRAM which is included in the interface hardware 72. Upon the completion of the execution, co-processor computer system 70 returns an acknowledgment through its operating system extension 74 and interface hardware 72 to provide an interrupt to operating system 36. This acknowledgement notifies operating system 36 that the requested function has been completed. Similarly, from the co-processor computer system side, a PC application program can run, for example, output data to be displayed on a video monitor. As discussed later, that data is then placed into the appropriate DPRAM location, particularly into the video RAM portion fo the DPRAM. That action of the co-processor system causes an interrupt to issue to the host computer system 66 which recognizes it as a video memory interrupt. In accordance with the issued interrupt the host system then accesses the appropriate location in DPRAM 76 and processes the video memory into the host computer system's video monitor 50.

The interface between host computer system 66 and co-processor computer system 70 provided by interface hardware 72 is interrupt driven. All functions in the computer systems are requested by interrupts. Hardware in interface hardware 72 automatically issues hardware interrupts to host CPU 12 or co-processor computer system 70 or host computer system 66, respectively. Furthermore, software applications can define new classes of events which allow either of the computer systems to use the facilities of the other, thereby creating an extended multiprocessor environment.

When data is to be exchanged between the two computer systems, it is transmitted via DPRAM 76. By using interrupts and a "lock byte" system discussed below to managa communications, both computer systems can operate completely independent of each other without degradation of performance.

In the preferred embodiment, predefined events occurring in co-processor computer system 70 which generate hardware interrupts to be acted upon by the host computer system can include requests by the co-processor computer system to access the color video memory of DPRAM 76, access the mono-chrome video portion of the DPRAM, access the mono CRT register, accessor the color CRT register, access the line printer register, access the serial data register, and read the keyboard register. Predefined events from the host computer system which force interrupts on the co-processor computer system, in the preferred embodiment, can include accessing the keyboard register, accessing the serial data register, and accessing the line printer register.

Concerning custom software defined interrupt events, they can be added to allow the programs on both sides of interface hardware 72 to communicate in any way desired. This could include, for example, code execution by one of the CPU's, including full access to all system functions of each operating system. The CPU of the co-processor computer system could, for example, create a task in the host computer system's multi-tasking environment and then instruct the task to execute host computer system library calls. The host computer system, in turn, can request software interrupts on the co-processor computer system.

If the reason for an interrupt is not one of the predefined events discussed above, the computer system which is being called on to take action by the other computer system must be informed with information on the appropriate function to undertake. These non-predefined events or logical events require the use of parameter blocks as discussed below.

It should be noted that in the above described operations, all function interrupts come in pairs, that is an initial interrupt when a function is requested by one of the CPUs and an acknowledgment interrupt from the other when the request has been satisfied. Thus, each CPU can issue a function request and then continue with other processing until the requested function is acknowledged. This allows completely asynchronous operation of both systems.

Interprocessor communication between CPU 12 of host computer system 66 and CPU 54 of co-processor computer system 70 is facilitated by the DPRAM connected to the CPU buses of the CPUs in both the host and the co-processor computer systems. The positioning of the DPRAM with respect to the host computer system and the co-processor system is shown in Fig. 7. Fig. 7 is identical to Fig. 2 with the exception that the DPRAM 76 and co-processor system 70 are connected into host computer system 66 as PC compatibility system 78. Specifically, DPRAM 76 is connected to both the Amiga bus and the CPU bus of the co-processor system. (Not all of the hardware of co-processor system 70 is illustrated in Fig. 7.)

In Fig. 8, details of DPRAM 76 are shown along with its association of the host and co-processor computer systems and their CPU buses. As shown in Fig. 8, the Amiga bus includes 68000 address bus 80 and 68000 data bus 82, and the co-processor bus includes the 8088 address bus 84 and 8088 data bus 86. DPRAM 76, in the preferred embodiment, is a 128K byte memory into which data is transferred from one of the computer systems for use by the other of the computer systems.

As also shown in Fig. 8, different portions of the DPRAM are reserved for specific uses. General purpose RAM area 88 is a 64K byte area used as a general purpose buffer; color video RAM 90 is a 32K byte area used for PC color video memory; and monochrome video RAM 92 is an 8K byte area for PC monochrome video memory.

In DPRAM 76, a 16K byte area is designated as a parameter RAM 94. Parameter RAM 94 provides two functions. First, both the host computer 66 and the co-processor computer 70 use one byte of parameter RAM 94 to assist the two systems in successfully sharing the DPRAM. Specifically, the byte is a "lock byte" used by either of the computer systems to signal that a new portion of general purpose RAM 88 is being alocated or freed. A check of this lock byte by one of the computer systems indicates whether the other of the computer systems is allocating data to a portion of the memory or is freeing a portion of the memory. If that is the case, the computer attempting to allocate or free a portion of the memory will wait until the other system is completed in its allocating or freeing and has so indicated that completion by configuring the lock byte to the "unused" status.

The second function of parameter RAM 94 is to hold blocks of parameters set by one computer which the other computer reads. For example, when host computer system 66 desires an action which is not one of the preferred events discussed above which automatically generates the appropriate hardware interrupt, for example, the transfer of a block of data to co-processor system 70, the transferring data is read into and held by general purpose RAM 88 of DPRAM 76 until that data is read out of that address by co-processor system 70. Parameter RAM 94 holds information necessary to perform the transfer, for example, the length and location of the data to be transferred and the desired destination of that data in the co-processor system, which information has been set by the computer requesting the transfer, in this instance, host computer 66.

Also included in DPRAM 76 is a final 8K byte area designated the I/O page RAM 96. It is used to perform various services such as mapping certain address ranges on the host computer system side of the interface hardware. On the co-processor system side of the hardware the I/O page RAM area contains a standard set of I/O register locations for the co-processor system that are used to control the serial and parallel ports and the PC monochrome and color CRT controller.

In the preferred system described herein concerning an Amiga computer system as host computer system 66 and the co-processor system 70 which creates a PC environment for runing PC application software, two support chips are provided to assist in the interprocessor communication achieved through the sharing of DPRAM 76. The details of those chips in interface hardware 72 are shown in Fig. 9. The two chips, called data bus translator ("DBT") 98 and address bus translator ("ABT") 100, are shown in block diagrammatic form connected to the CPU buses and control buses of both host computer system 66 and PC compatible co-processor computer system 70.

DBT 98 functions as a data bus transceiver for 68000 data bus 82 and for the 8088 data bus 86 to interface those data buses with DPRAM 76. More specifically, DBT 98 contains data translator 102 which functions to interface the 68000 CPU 12 of the host computer system to the DPRAM. Specifically, data translator 102 provides three transfer mechanisms. First, a word transfer operation is provided to realign bytes in accordance with individual processor requirements of the host and co-processor systems. Host CPU 12 stores the most significant byte at the low-order address while co-processor CPU 54 stores the most signficant byte at the high-order address. Data translator 102 provides this word transfer mechanism to make realignment automatic whenever data is written across interface hardware 72. Thus, this realignment places data in one of the computer systems into a format usable by the other system.

Second, data translator 102 provides byte transfer operations which transfer data straight across the interface hardware without the byte swap which occurs in the word transfer operation described above. Third, data translator 102 also provides a graphics transfer mechanism which is used to separate PC graphics data into discrete bit planes required by the host computer system. The graphics transfer mechanism substantially reduces the amount of software overhead necessary to display PC graphics on video monitor 50. A specific example of the graphics transfer mechanism is illustrated and discused in an article titled "The Commodore A2000" found at pages 84-98 of the March 1987 issue of BYTE magazine.

Transceiver 104 of DBT 98 functions as a high-speed data-transceiver. It is used to steer bidirectional data transfers from the 8-bit 8088 data bus 86 to the 16-bit DPRAM interface data bus 106. Auto config 108 of DBT 98 provides an autoconfiguration protocol which is used to insure that peripheral boards are installed into the system at boot-up. Auto config 108 automatically configures the co-processor system 70 into the host computer system 66 memory map during system boot. Finally, control 110 of DBT 98 functions to provide control signals for driving the various functions performed in DBT 98. Control 110 contains the base address comparator, address decoders, and other circuits which support the interface hardware operation.

The second chip in interface hardware 72 is ABT 100. It provides PC address translation, DPRAM arbitration, interrupt control logic, and keyboard emulation. Concerning address translation, ABT 100 translates co-processor system memory and I/O addresses into appropriate locations in the memory map of DPRAM 76 shown in Fig. 8. Any I/O request recognized by the ABT 100 as being I/O device emulated by the host computer system 66 triggers the translation function. For example, if host computer system 66 on the parallel port emulation and co-processor system 70 attempts to write to the printer data port, ABT 100 will generate the address for the printer data port's location in DPRAM 76. ABT 100 also maps various co-processor system memory requests into DPRAM 76, for example, video memory accesses to monochrome video RAM 92 or color video RAM 90 of the DPRAM show in Fig. 8.

Concerning the DPRAM arbitration, ABT 100 arbitrates access requests to DPRAM 76 between host computer system 66 and co-processor system 70. The two computer systems utilize independently generated clocks, thus requiring DPRAM requests to be synchronized to prevent conflict. Normally, the requests are serviced on a first-come, first-serve basis as discussed above. ABT 100 also provides DPRAM timing signals and DPRAM refresh signals as necessary.

With respect to the interrupt control logical function of ABT 100, when either one of the computer systems places information in DPRAM 76 as a predefined event, ABT 100 generates an interrupt to the other one of the computer systems as necessary. In the printer port example described above, an interrupt to host computer system 66 will occur to inform that system, by its reading of a register in ABT 100, that the printer data has been updated so that the host computer system 66 will take appropriate action on that data.

The last function of ABT 100 concerns keyboard emulation. Since the system described here uses keyboard 46 of the host system for data input to the co-processor system 70, the keyboard input from keyboard 46 must be translated into a keyboard equivalent for loading DPRAM 76 so that co-processor system 70 can utilize the information. Thus, another function of ABT 100 is to serialize the keyboard data received from host computer system 66 for use in co-processor system 70. Specifically, the Amiga host system loads an ABT register with a byte of keyboard data from keyboard 46. Thus, the ABT encodes the data serially for use by the co-processor system.

PC compatibility system 78 has been described in association with Fig. 9 for the host computer system 34. The system is connected to the CPU bus of the host computer system and allows true PC displays to be intermixed with all the displays available on the host computer system itself. The PC video information can be displayed in standard windows of the multi-tasking host computer system. Additionally, all PC video modes are supported, that is monochrome and color text and color graphics, which can be displayed simultaneously in different windows on host video monitor 50.

In the system described above, the video portion of the co-processor system memory is mapped directly into monochrome video RAM 92 or color video RAM 90 of DPRAM 76. That writing is a predetermined event and is performed by the co-processor system BIOS screen I/O routines. Thus, the co-processor system action causes the transmission of a hardware generated interrupt signal to the host computer system which informs that computer system that there has been activity in the video memory portion, that is, portion 90 or 92 of DPRAM 76, and the host system acts accordingly on that data.

Since this mapping is transparent to co-processor system 70, the video display is completely compatible both to applications that use the ROM BIOS in the co-processor system and to applications that word directly with the with the video memory in that system. Meanwhile, software running on the host system 66 can use the video data mapped directly into DPRAM 76 to create windows on Amiga video monitor 50 that emulate faithfully video generated in co-processor system 70. By using two different portions of the DPRAM for video memory, that is monochrome video RAM 92 and color video RAM 90, both monochrome and color graphics can be emulated at the same time and shown on the video monitor.

In addition to the video display enhancement provided by the system, due to the interface hardware and supporting software, the host system's keyboard 46 and serial and parallel ports 24, 26 can be used by the PC co-processor system as discussed above.

To further assist in explaining the method of the present invention, two specific examples of typical communications between the CPUs of the host and co-processor systems are provided. They are given only for exposition and are not to be taken as limiting the claimed invention.

In a first example, the method of communicating a monochrome or color video display from co-processor system 70 to video monitor 50 on the Amiga host system 66 is discussed. In this example, a PC "Flight Simulator" program is running on the co-processor system which is to be displayed in the video monitor on the host system. When the "Flight Simulator" program outputs data to be read on a video monitor, the co-processor system using its BIOS screen I/O routines as discussed above writes the information into the monochrome video memory 92 or into color video memory 90 of DPRAM 76 as determined by the specific output data.

The computer system requesting the operation, that is co-processor system 70, has therefore requested a function to be performed and has transferred the data to the DPRAM. Since the function desired by the requesting computer is one of the predefined events a hardware interrupt issues to the other one of the two computer systems, that is, host computer system 66 in this example. The interrupt indicates to the host computer system the type of function requested, that is, a video memory request. At that time the host computer system acting on the informative interrupt accesses the data in the appropriate video RAM portion of DPRAM 76 and acts on the data contained therein. For example, the host system updates the PC video display shown on host computer system video monitor 50. The host system, upon completion of the requested function, informs the requesting computer system (in this instance, co-processor system 70) by way of acknowledgment that the requested function has been completed. That acknowledgment initates the erasing of the data temporarily stored on the DPRAM.

An example of the interprocessor communication using parameter RAM 94 of DPRAM 76 is as follows. In this example, the requesting computer system is the host computer system which wishes to write a block of data into the specific portion of the memory in co-processor system 70. Also in this instance, the specific function is a "peek/poke" memory function. When the peek/poke function is requested in host computer system 66, the host computer system generates data representing the requested function and, in this instance, would request allocation of memory in parameter RAM 94 and in general purpose RAM 88 of DPRAM 76. Then, the host system provides the data on how the function is to be performed to DPRAM 76. Specifically, the data would include the address in the co-processor system to be written, how many bytes of data are to be transferred, and where in the general purpose RAM the actual data is stored. The actual data is then transferred from the host system into the designated portion of general purpose RAM 88. That data is temporarily stored in the DPRAM.

At this time, the host system calls the peek/poke memory function and issues an interrupt to the interrupt driven co-processor system 70. When the interrupt is received by the co-processor system, that system will be informed that the host system desires the transfer of data to the co-processor system; and, therefore, the co-processor system takes action to examine parameter RAM 94 for data on the specific function requested.

Using that data, the co-processor system locates in and transfers the data from general purpose RAM 88 to the designated final location within the co-processor system. After that data has been transferred, the co-processor generates and transfers an acknowledgment of completion to the host system. At that time, the memory in DPRAM 76 allocated for the specific function is cleared so that the memory will be available for other tasks.

In the above described system, a method is provided to allow the high performance host computer system to utilize PC application software as desired. Thus, the user of the host system can enjoy the available library of PC application programs which otherwise would be unavailable. Furthermore, the user can process the PC programs in a multi-tasking environment using enhanced graphics made capable by the host system.

It will become apparent to those skilled in the art that various modifications and variations can be made in the method of communicating data between the CPU of the host computer system and the CPU of a co-processor computer system of the present invention without departing from the scope or spirit of the invention. Thus, it is intended that the present invention covers the modifications and variations of this invention provided they come within the scope of the intended claims and their equivalence.

WHAT IS CLAIMED IS

1. A method of communicating data between the central processing unit (CPU) of a host computer system and the CPU of a co-processor computer system, the method using a dual port memory connected to the data and address buses of the CPUs and comprising the steps of:

requesting in one of the two computer systems one of a plurality of functions to be performed;

generating, using the requesting computer system, parameter data representing the requested function;

transferring, using the requesting computer system, the parameter data to the dual port memory;

storing the parameter data in the dual port memory;

issuing, using the requesting computer system, an interrupt to the other one of the two computer systems, the interrupt indicating to the other one of the two computer systems the type of function requested; and

accessing, using the other one of the two computer systems, the parameter data in the dual port memory based on the interrupt received to perform the requested function.

2. The method of claim 1 further comprising the steps of:

issuing to the requesting computer system at the completion of the requested function, using the other one of the two computer systems, an acknowledgment of the completed requested function and

erasing the stored data from the dual port memory, using the other one of the two computer systems.

3. The method of claim 1 wherein the step of transferring the parameter data to the dual port memory further includes the step of:

transferring into the dual port memory using the requesting computer system, transfer data desired to be transferred from the requesting computer system to the other one of the computer systems; and

the step of accessing the parameter data in the dual port memory further includes the steps of

accessing in the dual port memory the transfer data and

acting upon the transfer data in accordance with the requested function using the other one of the computer systems.

4. The method of claim 1 further including the step of processing requests for access to the dual port memory made by the CPUs one at a time on a first-come, first-serve basis.

5. The method of claim 3 further including the step of translating the transfer data issued by one of the two computer systems into a data format usable by the other of the two computer systems into a data format usable by the other of the two computer systems before the transfer data is stored in the dual port memory.

6. A method of communicating data between the central processing unit (CPU) of a host computer system and the CPU of a co-processor computer system, the method using a dual port memory connected to the data and address buses of the CPUs and comprising the steps of:

requesting in one of the two computer systems one of a plurality of predefined functions to be performed;

generating, using the requesting computer system, data representing the requested function;

transferring, using the requesting computer system, the data to the dual port memory;

storing the data in the dual port memory;

issuing, using the requesting computer system, an interrupt to the other one of the two computer systems, the interrupt indicating to the other one of the two computer systems the type of function requested; and

accessing, using the other one of the two computer systems, the data in the dual port memory based on the interrupt received to perform the requested function.

7. The method of claim 6 further comprising the steps:

issuing to the requesting computer system at the completion of the requested function, using the other one of the two computer systems, an acknowledgment of the completed requested function and erasing the stored data from the dual port memory, using the other one of the two computer systems.

8. The method of claim 1 further including the step of processing requests for access to the dual port memory made by the CPUs one at a time on a first-come, first-serve basis.

9. The method of claim 6 including the step of translating the data issued by one of the two computer systems into the data format usable by the other of the two computer systems before the data is temporarily stored in the dual port memory.

FIG. 1

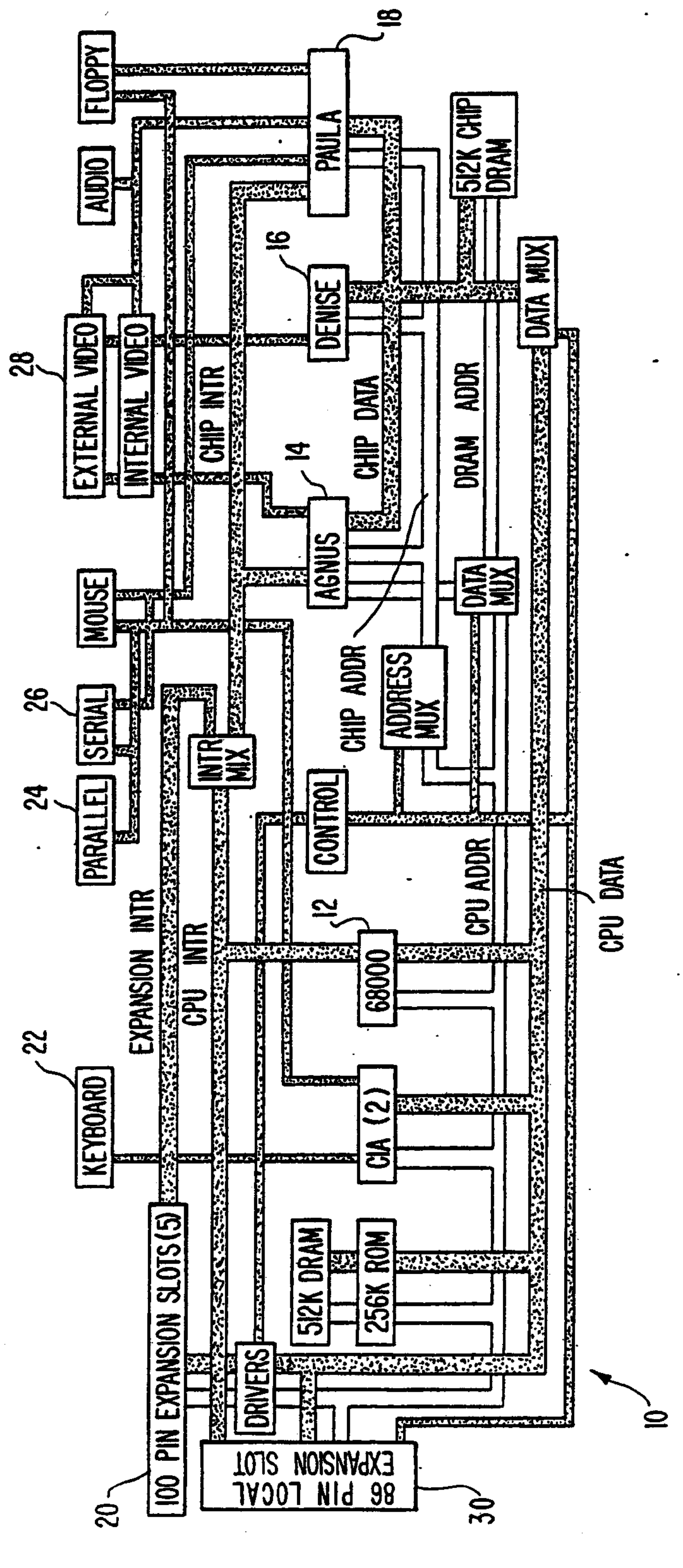

FLOPPY
AUDIO
EXTERNAL VIDEO
INTERNAL VIDEO
28
MOUSE
SERIAL
26
PARALLEL
24
KEYBOARD
22
100 PIN EXPANSION SLOTS (5)
20
86 PIN LOCAL EXPANSION SLOT
30
PAULA
18
512K CHIP DRAM
DENISE
16
DATA MUX
CHIP INTR
CHIP DATA
DRAM ADDR
AGNUS
14
DATA MUX
ADDRESS MUX
CHIP ADDR
INTR MIX
CONTROL
CPU ADDR
CPU DATA
EXPANSION INTR
CPU INTR
68000
12
CIA (2)
512K DRAM
256K ROM
DRIVERS
10

FIG. 3

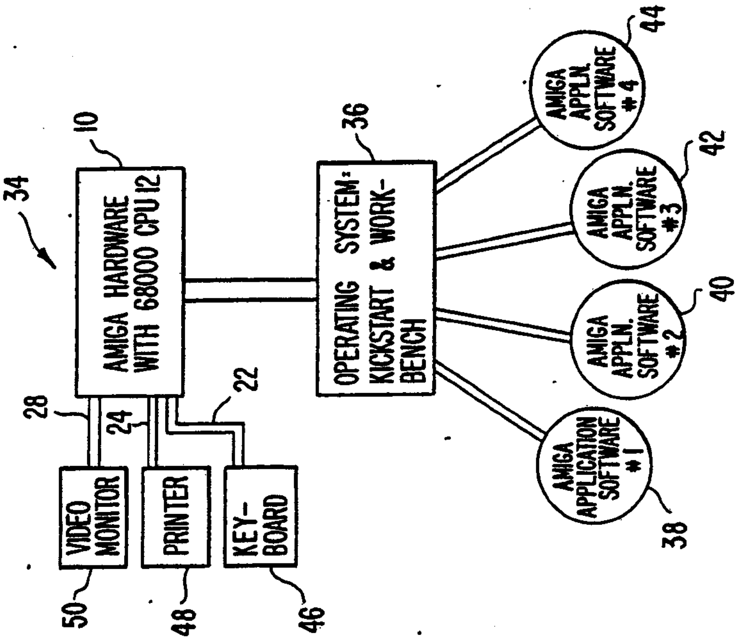

34
AMIGA HARDWARE WITH 68000 CPU 12
10
OPERATING SYSTEM: KICKSTART & WORK-BENCH
36
28
24
22
VIDEO MONITOR
50
PRINTER
48
KEY-BOARD
46
AMIGA APPLICATION SOFTWARE #1
38
AMIGA APPLN. SOFTWARE #2
40
AMIGA APPLN. SOFTWARE #3
42
AMIGA APPLN. SOFTWARE #4
44

FIG. 2

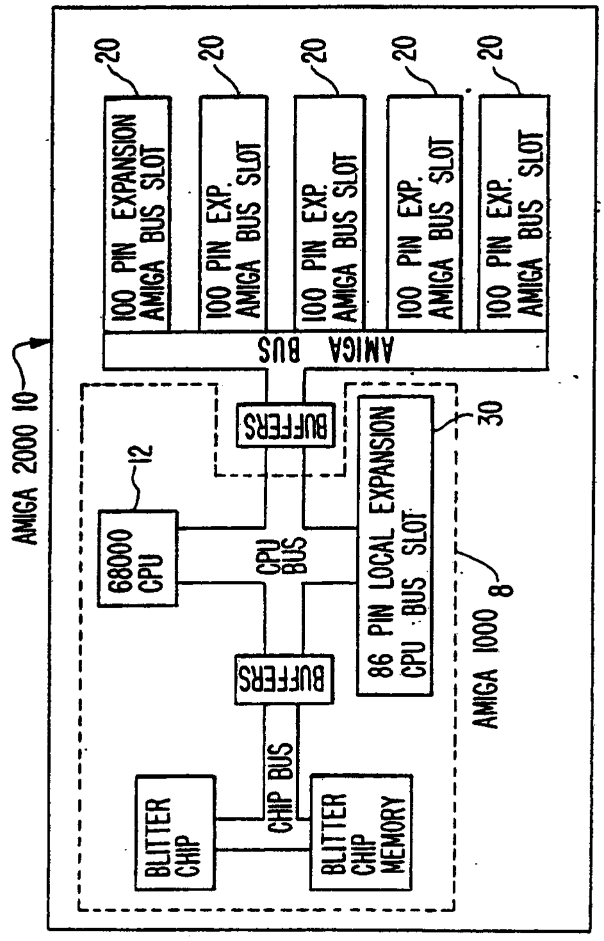

AMIGA 2000 10
100 PIN EXPANSION AMIGA BUS SLOT
20
100 PIN EXP. AMIGA BUS SLOT
20
100 PIN EXP. AMIGA BUS SLOT
20
100 PIN EXP. AMIGA BUS SLOT
20
100 PIN EXP. AMIGA BUS SLOT
20
AMIGA BUS
BUFFERS
68000 CPU
12
CPU BUS
86 PIN LOCAL EXPANSION CPU BUS SLOT
30
AMIGA 1000 8
BUFFERS
CHIP BUS
BLITTER CHIP
BLITTER CHIP MEMORY

FIG. 4

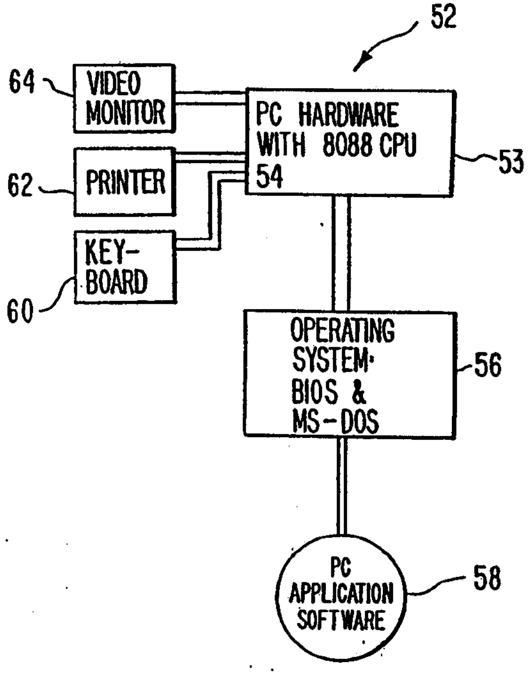

52
64
VIDEO MONITOR
PC HARDWARE WITH 8088 CPU
54
53
62
PRINTER
60
KEY-BOARD
OPERATING SYSTEM: BIOS & MS-DOS
56
PC APPLICATION SOFTWARE
58

FIG. 6

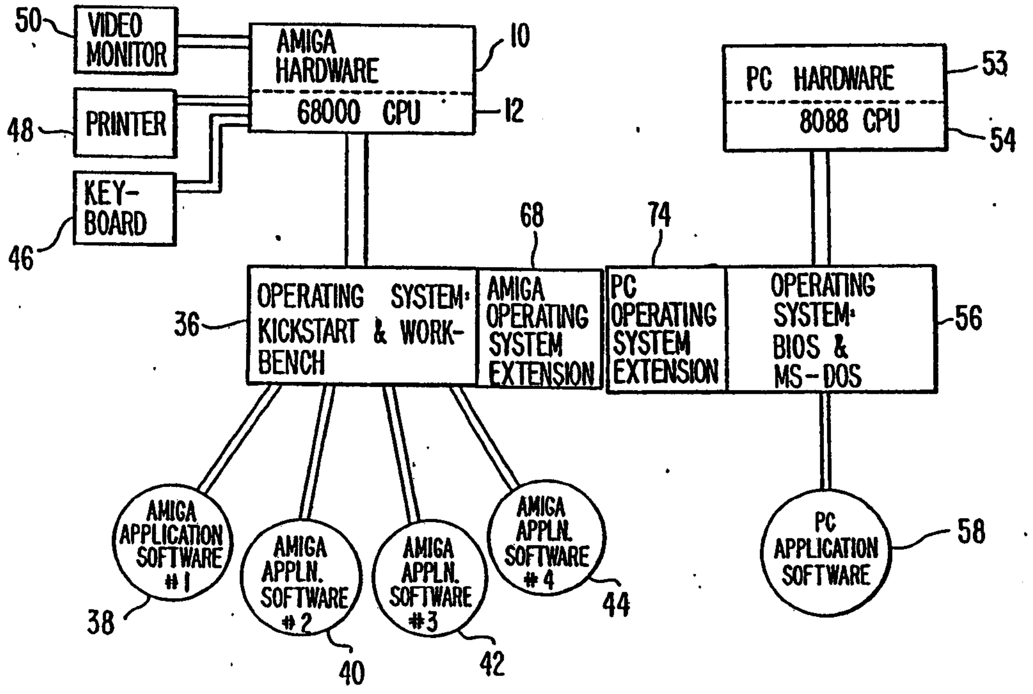

50
VIDEO MONITOR
AMIGA HARDWARE
10
68000 CPU
12
48
PRINTER
46
KEY-BOARD
PC HARDWARE
53
8088 CPU
54
68
74
36
OPERATING SYSTEM: KICKSTART & WORK-BENCH
AMIGA OPERATING SYSTEM EXTENSION
PC OPERATING SYSTEM EXTENSION
OPERATING SYSTEM: BIOS & MS-DOS
56
AMIGA APPLICATION SOFTWARE #1
38
AMIGA APPLN. SOFTWARE #2
40
AMIGA APPLN. SOFTWARE #3
42
AMIGA APPLN. SOFTWARE #4
44
PC APPLICATION SOFTWARE
58

FIG. 5

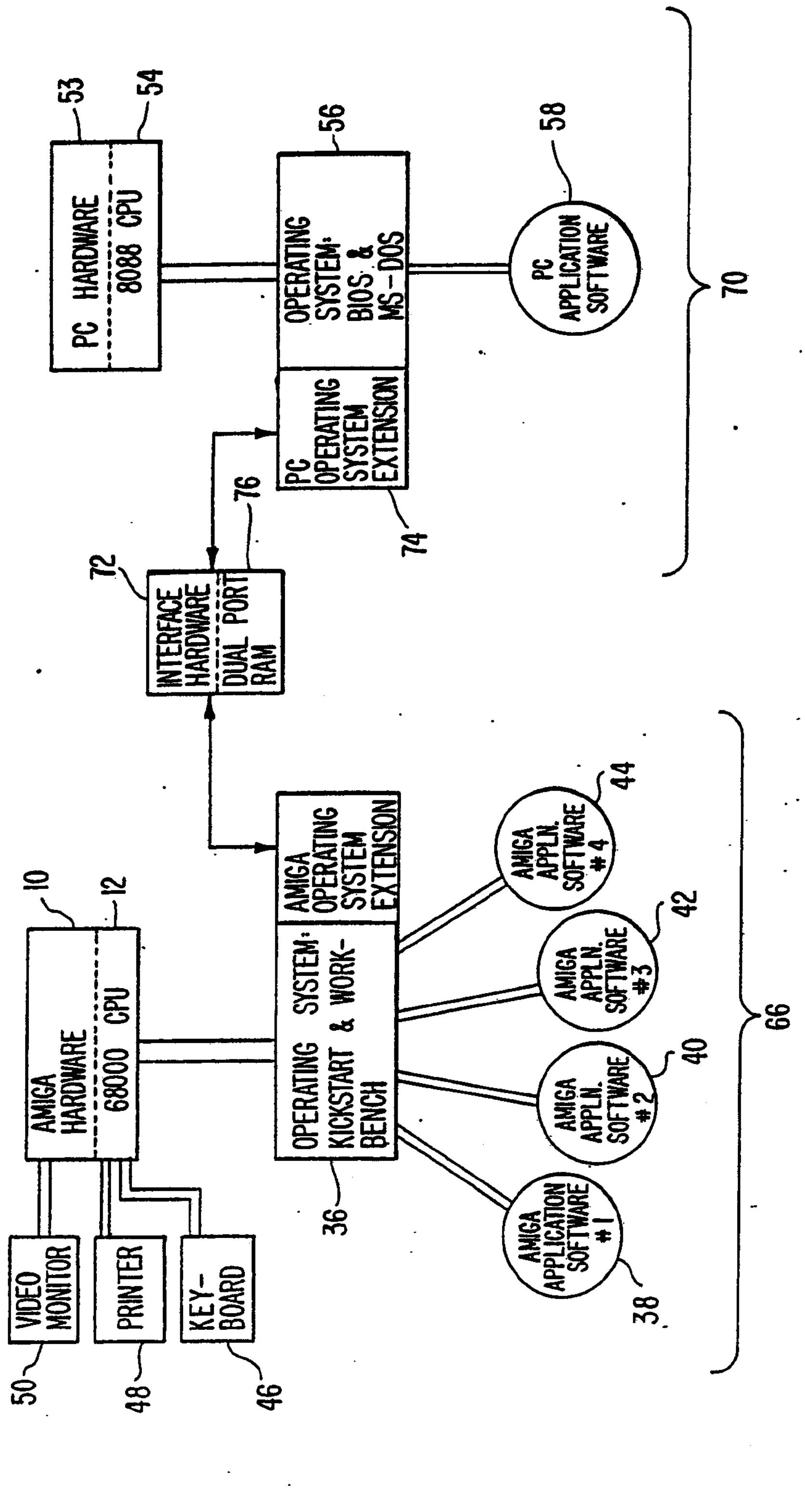

PC HARDWARE
8088 CPU
53
54
OPERATING SYSTEM: BIOS & MS-DOS
56
PC APPLICATION SOFTWARE
58
PC OPERATING SYSTEM EXTENSION
74
76
70
INTERFACE HARDWARE
DUAL PORT RAM
72
AMIGA HARDWARE
68000 CPU
10
12
OPERATING SYSTEM: KICKSTART & WORK-BENCH
36
AMIGA OPERATING SYSTEM EXTENSION
AMIGA APPLICATION SOFTWARE #1
38
AMIGA APPLN. SOFTWARE #2
40
AMIGA APPLN. SOFTWARE #3
42
AMIGA APPLN. SOFTWARE #4
44
66
VIDEO MONITOR
50
PRINTER
48
KEY-BOARD
46

FIG. 7

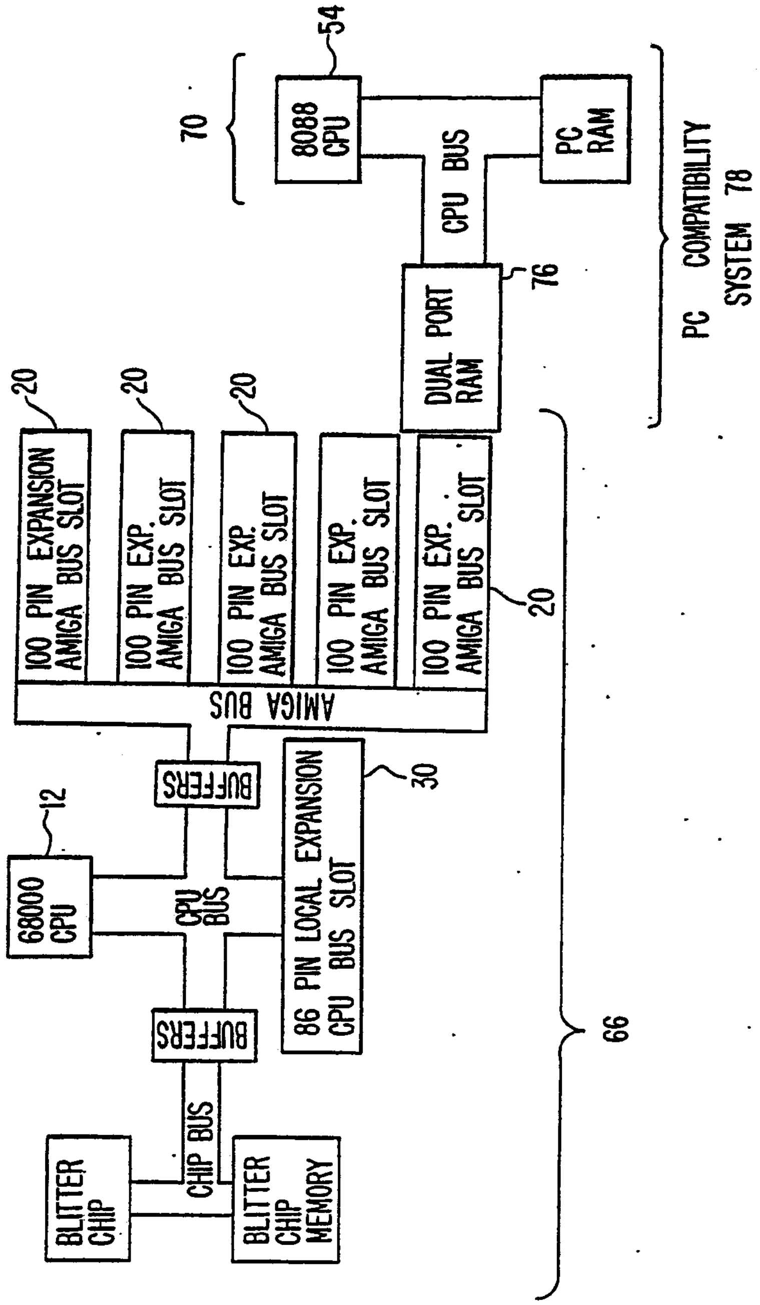

BLITTER CHIP
BLITTER CHIP MEMORY
CHIP BUS
BUFFERS
68000 CPU
12
CPU BUS
BUFFERS
86 PIN LOCAL EXPANSION CPU BUS SLOT
30
AMIGA BUS
100 PIN EXPANSION AMIGA BUS SLOT
100 PIN EXP. AMIGA BUS SLOT
100 PIN EXP. AMIGA BUS SLOT
100 PIN EXP. AMIGA BUS SLOT
100 PIN EXP. AMIGA BUS SLOT
20
20
20
20
66
DUAL PORT RAM
76
CPU BUS
8088 CPU
54
PC RAM
70
PC COMPATIBILITY SYSTEM 78

FIG. 8

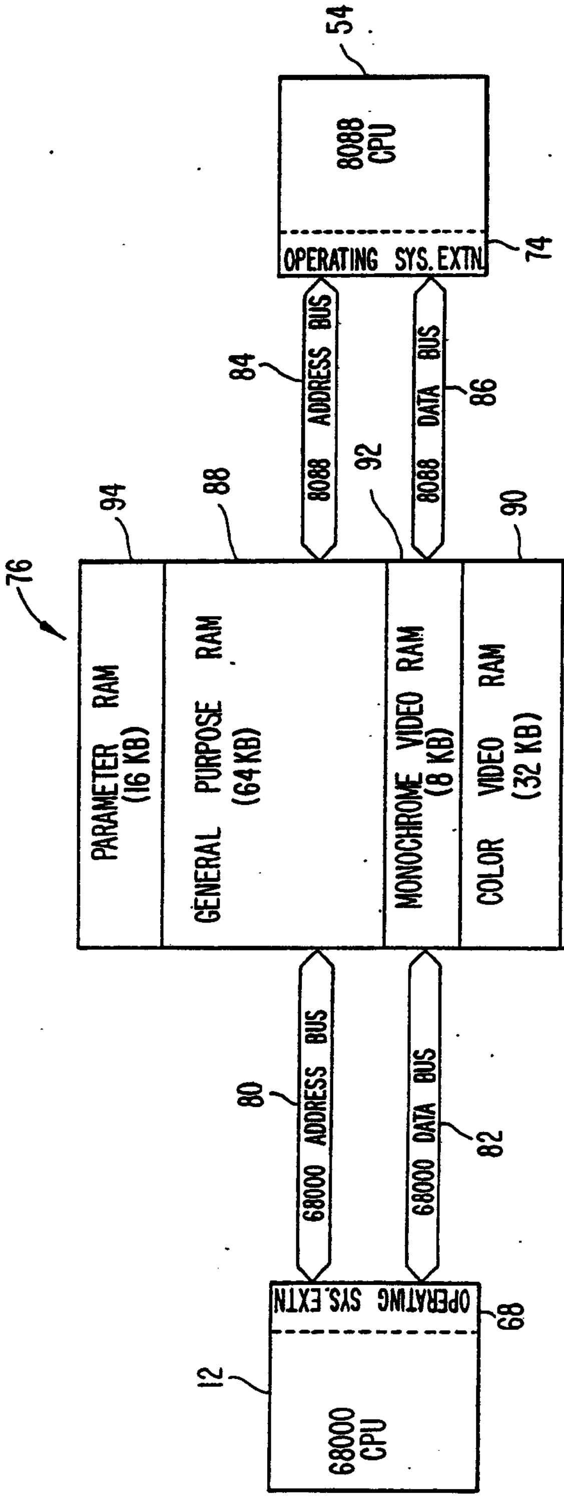
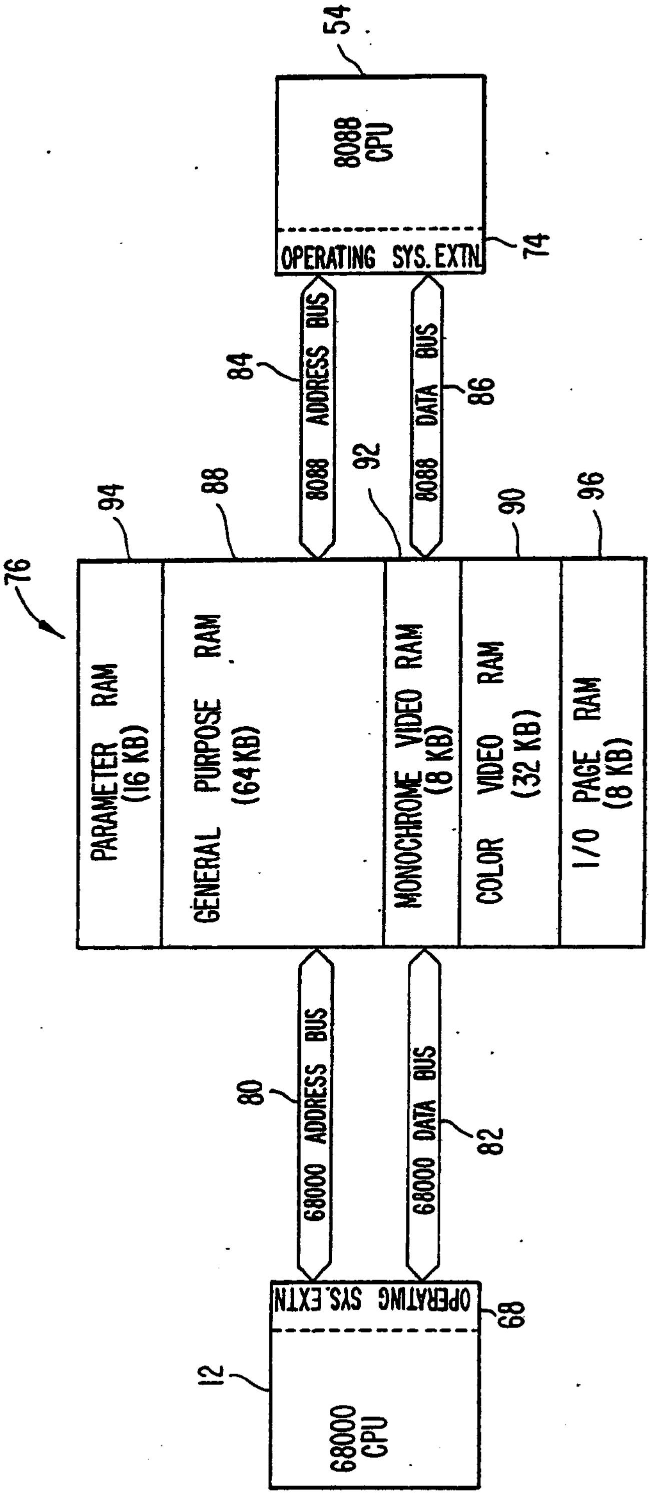

54
8088 CPU
OPERATING SYS. EXTN
74
84
8088 ADDRESS BUS
8088 DATA BUS
86
94
88
92
90
96
76
PARAMETER RAM (16 KB)
GENERAL PURPOSE RAM (64 KB)
MONOCHROME VIDEO RAM (8 KB)
COLOR VIDEO RAM (32 KB)
I/O PAGE RAM (8 KB)
80
68000 ADDRESS BUS
68000 DATA BUS
82
OPERATING SYS. EXTN
68
12
68000 CPU

FIG. 8


54
8088 CPU
OPERATING SYS. EXTN
74
84
8088 ADDRESS BUS
8088 DATA BUS
86
76
94
88
92
90
96
PARAMETER RAM (16 KB)
GENERAL PURPOSE RAM (64 KB)
MONOCHROME VIDEO RAM (8 KB)
COLOR VIDEO RAM (32 KB)
I/O PAGE RAM (8 KB)
80
68000 ADDRESS BUS
68000 DATA BUS
82
OPERATING SYS. EXTN
68
12
68000 CPU

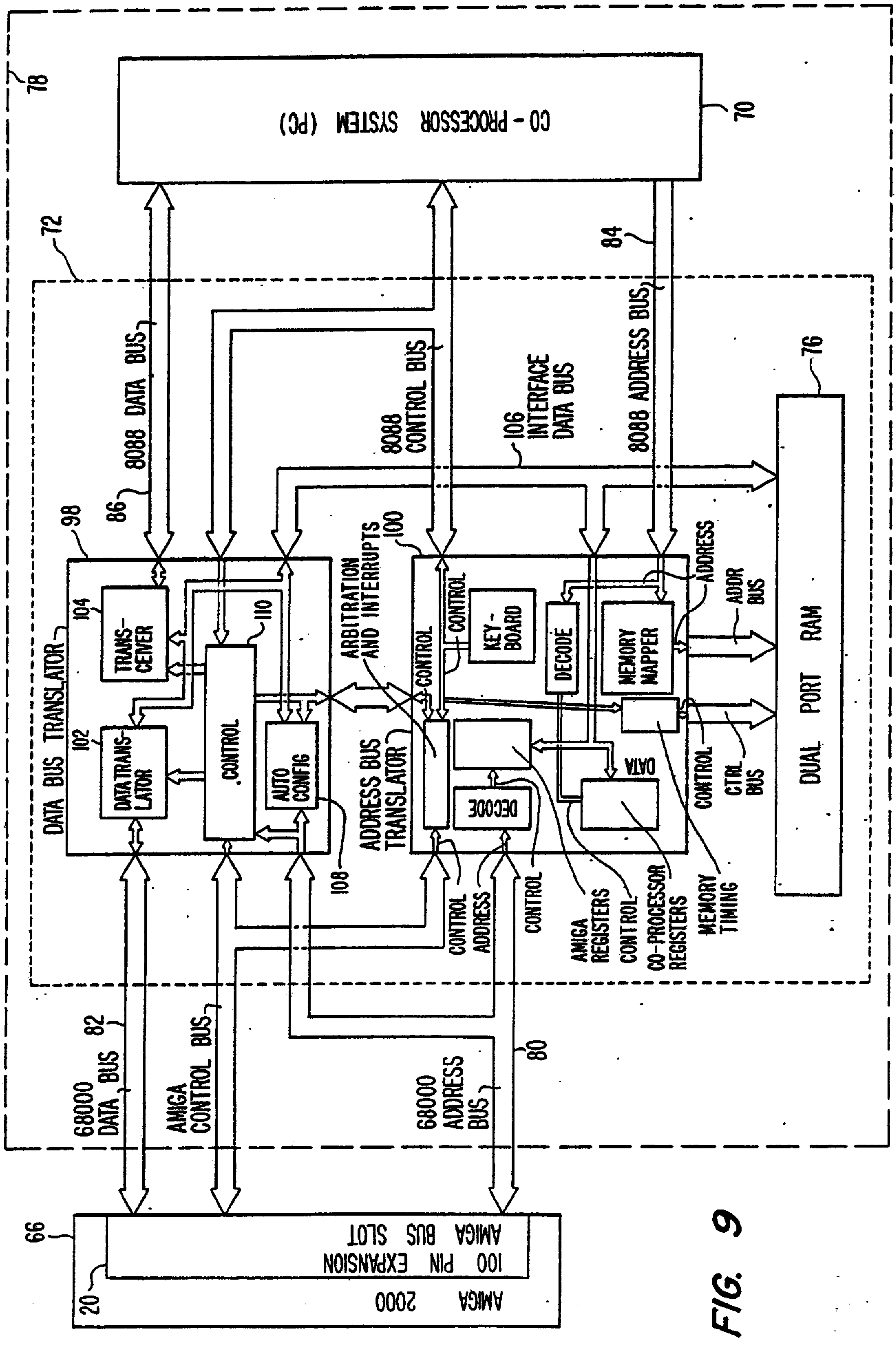

78
CO - PROCESSOR SYSTEM (PC)
70
72
84
8088 DATA BUS
8088 CONTROL BUS
106 INTERFACE DATA BUS
8088 ADDRESS BUS
76
86
98
100
DATA BUS TRANSLATOR
104
TRANS-CEIVER
110
102
DATA TRANS-LATOR
CONTROL
AUTO CONFIG
108
ARBITRATION AND INTERRUPTS
ADDRESS BUS TRANSLATOR
CONTROL
CONTROL
KEY-BOARD
DECODE
MEMORY MAPPER
ADDRESS
ADDR BUS
DATA
DECODE
CONTROL
CTRL BUS
DUAL PORT RAM
CONTROL
ADDRESS
CONTROL
AMIGA REGISTERS
CONTROL
CO-PROCESSOR REGISTERS
MEMORY TIMING
68000 DATA BUS
82
AMIGA CONTROL BUS
68000 ADDRESS BUS
80
66
100 PIN EXPANSION AMIGA BUS SLOT
20
AMIGA 2000

FIG. 9

INTERNATIONAL SEARCH REPORT

International Application No. PCT/US88/001166

I. CLASSIFICATION OF SUBJECT MATTER (if several classification symbols apply, indicate all) [6]

According to International Patent Classification (IPC) or to both National Classification and IPC

IPC (4): G06F 3/00 5/00 13/18

U.S. Cl. 364/200

II. FIELDS SEARCHED

Minimum Documentation Searched [7]

Classification System	Classification Symbols
U.S.	364/200, 900

Documentation Searched other than Minimum Documentation to the Extent that such Documents are Included in the Fields Searched [8]

III. DOCUMENTS CONSIDERED TO BE RELEVANT [9]

Category *	Citation of Document, [11] with indication, where appropriate, of the relevant passages [12]	Relevant to Claim No. [13]
X	US, A, 4,215,399 (PAVICIC ET AL.) 29 JULY 1980 See entire document.	1-9
X	US, A, 4,547,849 (LOUIE ET AL.) 15 OCTOBER 1985 See entire document.	1-9
X,P	US, A, 4,663,730 (IKEDA) 5 MAY 1987 See entire document.	1-9
X,P	US, A, 4,679,166 (BERGER ET AL.) 7 JULY 1987 See entire document.	1-9
Y,P	US, A, 4,722,048 (HIRSCH ET AL.) 26 JANUARY 1988 See entire document.	1, 4, 6, 8
Y,P	US, A, 4,731,736 (MOTHERSOLE ET AL.) 15 MARCH 1988 See entire document.	1, 3, 6
A,P	US, A, 4,695,945 (IRWIN) 22 SEPTEMBER 1987 See entire document.	1,6
A	US, A, 4,594,657 (BYRNS) 10 JUNE 1986 See entire document.	1,6

* Special categories of cited documents: [10]

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

IV. CERTIFICATION

Date of the Actual Completion of the International Search 21 JUNE 1988	Date of Mailing of this International Search Report 01 AUG 1988
International Searching Authority ISA/US	Signature of Authorized Officer David L. Clark DAVID L. CLARK