# US-4974159-A Method of transferring control in a multitasking computer system

(1) TECHNICAL FIELD  
(2) This invention relates multitasking computer systems and methods, more particularly, to a computer method and system for transferring control in a multitasking environment.  
(3) BACKGROUND ART  
(4) Newly designed microprocessors may include enlarged memory addressing facilities and revised architecture which result in enhanced capabilities. When such microprocessors are used in new computer systems, they often produce computers which are functionally superior to their predecessors due to these enhanced capabilities. Despite any functional advantages a new computer may have over its predecessors, a computer employing an improved microprocessor may not be a commerical success. Computer programs, sometimes referred to as "software," are microprocessor specific. Therefore, when a computer employing a new microprocessor is introduced into the marketplace, there is generally little or no software which can run on it. Existing software, written for previous microprocessors, is incompatible with the new computer. As a result, sales of such new computers will often be sluggish until consumers see that adequate software is available for the computer. Additionally, consumers with libraries of software for existing computers may be reluctant to purchase new computers which would require them to invest in all new software. This problem is often compounded by the fact that software writers and publishers are reluctant to produce software for a new microprocessor until sales of computers incorporating the microprocessor are sufficient to create a relatively large group of potential purchasers of the software. This "wait and see" attitude on the part of both consumers and software writers can jeoparize the success of a new microprocessor and computers using the microprocessor.  
(5) Designers of new microprocessors sometimes attempt to solve this problem by designing a new microprocessor such that it will operate in multiple modes. In a first mode, for example, the microprocessor will emulate a prior microprocessor and run existing programs written for the prior microprocessor. In a second mode, the microprocessor will make full use of its enhanced capabilities. Such a design will enable manufacturers of computer systems using the microprocessor to advertise that the entire body of existing programs written for the prior microprocessor will run on their computer, thereby (in theory) stimulating computer sales to a point where software writers will begin to write programs designed to run in the new enhanced mode.  
(6) One such microprocessor is the Intel 80286, which is manufactured by the Intel Corporation of Santa Clara, Calif. The design and operation of the Intel 80286 is described in detail in a publication entitled "iAPX286 Programmer's Reference Manual Including the iAPX286 Numeric Supplement," which is available from the Intel Corporation and is hereby incorporated by reference.  
(7) The Intel 80286 (hereinafter "80286") operates in two modes. In a first mode, called the "real mode," the 80286 emulates the architecture of Intel's previous 8086, 8088 microprocessor family, which is used in the IBM PC and compatible computers, for example. Thus, computers which incorporate the 80286 microprocessor, such as the IBM PC/AT, can run existing 8086 programs written for the IBM PC and compatible computers.  
(8) In a second mode, called the "protected mode," the 80286 architecture provides enlarged memory addressing capability, enhanced multitasking support features, and a sophisticated protection scheme.  
(9) Another such microprocessor is the Intel 80386. The design and operation of the Intel 80386 is described in detail in a publication entitled "iAPX386 Programmer's Reference Manual Incuding the iAPX386 Numeric Supplement", which is available from the Intel Corporation and is hereby incorporated by reference.  
(10) The 80386, in addition to a real and protected mode as described above for the 80286, has a third mode, called virtual-8086 mode. In virtual-8086 the 80386 emulates the 8086 processor in a manner similar to the real mode. The distinction between real and virtual-8086 mode is that in virtual-8086 mode the 80386 provides memory-management, protection, and multitasking support. The virtual-8086 mode allows 8086 programs to execute as a task on the 80386. Each task in virtual-8086 mode has the illusion that it is executing on a 8086.  
(11) A virtual machine monitor (VMM), which is special operating-system software, coordinates the multitasking of several 8086 programs. The VMM executes in protected mode. There are two standard techniques for transferring control from a task to a VMM so that another task can be started. First, the VMM configures the 80386 so that all interrupts, software and hardware, that are executed by an 8086 program cause control to be transferred to the VMM. Second, the VMM sets a timer interrupt. When interrupted after the specified interval, the VMM receives control.  
(12) These techniques can be used to support the transfer from 8086 programs that are designed to execute under a disk operating system (DOS). A typical personal computer DOS, such as MS-DOS Version 3.X offered by Microsoft Corporation of Redmond, Washington, is a single-threaded operating system; that is, the DOS is not designed to support a multitasking environment. When the DOS is executed in a multitasking environment, problems can occur when the DOS is interrupted. If the DOS is randomly interrupted during the execution of a function, as an interval timer, then the transfer to another task can cause the DOS data structures to be corrupted. Consequently, a VMM will allow the DOS to complete a function before another task is started. Upon exit from a DOS system call, the DOS data structures are in an appropriate state.  
(13) Unfortunately, a VMM that allows all functions calls to complete before transferring tasks will result in poor system performance. Several system calls of DOS may take an indefinite amount of time to complete. For example, a call to retrieve a character from the keyboard will not complete until a key is actually entered. Similarly, if a program issues a system call to read from a communication port, the system call will not complete until a character is actually received. The DOS loops through a section of code checking the port to see if the character has been received. Consequently, no other task can be scheduled for this indefinite period of time.  
(14) A timer interrupt of system calls of indefinite duration is insufficient because the VMM does not know whether the DOS was in a state where the data structures were non-corruptable.  
(15) SUMMARY OF THE INVENTION  
(16) It is an object of the present invention to provide an improved method and system for transferring control in a multitasking computer system.  
(17) It is a further object of the present invention to provide such a method and system that will avoid inefficiencies and poor system performance caused by system calls of indefinite duration.  
(18) These and other objects, which will be apparent as the invention is more fully described below, are obtained by providing an improved method and system. In preferred embodiments for use with 80386 machines, the virtual machine monitor (VMM) writes a virtual machine break point (VMBP) instruction into the executable code of selected DOS routines to cause the DOS routines to transfer control to the VMM. By selecting systems calls of indefinite duration for the application of this method, system performance can be improved. In this preferred embodiment for 80386 machines, 8086 programs designed to run under DOS can be more efficiently executed in a multitasking environment.  
(19) BRIEF DESCRIPTION OF THE DRAWINGS  
(20) FIG. 1 is an illustration of the VDMM with the BPL addresses loaded in accordance with a preferred embodiment.  
(21) FIG. 2 is an illustration of BPL Address Table and Simulation Code Table in accordance with a preferred embodiment.  
(22) FIG. 3 is an illustration of the Interrupt Descriptor Table set up in accordance with a preferred embodiment of the invention.  
(23) FIG. 4 is an illustration of the ARPL instruction used as the BPL in a preferred embodiment of the invention.