SYSTEM AND METHOD FOR UPGRADING SOFTWARE

FIELD OF THE INVENTION

The invention relates generally to the upgrading of software in computing devices. More particularly, the present invention is well suited to the task of upgrading operating systems software in non-volatile memory.

BACKGROUND OF THE INVENTION

Generally, computing devices include some form of non-volatile memory containing a software program that controls the start up and initialization of the computing device. For computing devices such as personal computers, the software program contained in the non volatile memory controls the transfer of an operating system software program from a high capacity non-volatile storage device to a volatile storage device (ie usually from the Hard Disk Drive to Random Access Memory).

However, in smaller computing devices that do not include high capacity non-volatile storage devices, such as embedded systems or Personal Digital Assistants (PDA), the operating system software is usually stored in the non-volatile memory. Generally, operating system software for these types of devices is stored in "flash memory" which is a form of non-volatile memory that will not be erased when power is disconnected from the device.

With reference to Figure 1, a diagrammatic representation of the basic memory processing components of a relatively small computing device is provided. Figure 1 details a Micro Controller (MCU)(1O), a Random Access Memory unit (RAM)(15) and a Flash Read Only Memory (ROM)(20) each of these components communicating with each other via the data and control bus (25). Software programs that control the start up and initialization of computing devices that do not include a high capacity non-volatile storage device are usually stored in the Flash ROM (20).

The integrity of software programs stored in the non-volatile memory is critical as it is the first program that is executed when the computing device is initialized. When the software program stored in Flash ROM (20) is an operating system, it is likely that upgrades to the operating system software will be released

during the operational life of the computing device. In this instance, the "old" operating system in the Flash ROM needs to be replaced.

The process of upgrading an operating system is problematical as errors during an upgrade process could cause the computing device to become inoperable. With known systems and methods for upgrading an operating system in a computing device, if an error occurs during the upgrade process it is usual for the owner of the computing device to return the device to a service centre for reprogramming.

The usual technique for upgrading an operating system in a small computing device is described with reference to Figure 2. Usually, the computing device is connected to a host device which includes a data file representing the "new" operating system. Having established a data communication link between the host device and the computing device, the new operating system is first copied into RAM (30). The process of transferring the new operating system into the RAM (30) is controlled by the MCU (35). Having completed this transfer, a new operating system image resides within the RAM (30) and the computing device then transfers the new operating system image from the RAM (30) to the Flash memory (40) thus overwriting the old operating system. At this stage, the computing device is re-initialized and executes the new operating system.

Generally, the time required to copy the new operating system image from RAM (30) to Flash ROM (40) is relatively long. During this time, if any errors occur the computing device is usually rendered inoperable.

There are various proposals to overcome this problem such as that described in U.S. Patent specification No. 6,205,513 (Godicke). In the proposal of Godicke, a boot program is copied into RAM that has a battery back-up facility. When a power failure is detected, sufficient power remains to write the RAM entry point into Flash ROM. When power is returned to the device, program execution is directed to that RAM entry point.

However, the proposal of Godicke requires several additional components in the computing device. For example, RAM with a battery back-up facility is required with sufficient reserve power for the device to allow an entry point in the RAM to be written to Flash Memory. Further, the proposal of Godicke requires a computing device to include power failure detection circuitry. Inclusion of these

additional components necessarily increases the cost of the computing device. Further, the proposal described by Godicke is only directed to overcoming the problem of a power failure during the process of upgrading an operating system. The method described by Godicke will not overcome other types of problems during the upgrade of an operating system such as a user inadvertently resetting the computing device during the upgrade process or inadvertently disconnecting the communications link between the computing device and a host device.

Another proposal is described in patent specification No. WO 01/07922 (Tallam) which includes the provision of two operating systems, a primary and recovery operating system, such that the computing device may revert to the recovery operating system in the event that a problem occurs. In addition to storing a primary and recovery operating system, the proposal of Tallam requires a flag or some other means to direct the computing device to execute one or other of the stored operating systems.

The proposal described by Tallam is particularly disadvantageous for computing devices where memory space is a valuable resource as the recovery operating system always resides in memory and consumes memory space irrespective of whether it is required or not. Further, the recovering operating system remains in the computing device and is not upgraded over time. This is particularly disadvantageous if there is a requirement to revert to an alternative data transfer communication protocol sometime in the future. For these reasons, the proposal of Tallam is not considered to be feasible particularly for computing devices such as personal digital assistants where memory is a limited resource.

Any discussion of documents, devices, acts or knowledge in this specification is included to explain the context of the invention. It should not be taken as an admission that any of the material formed part of the prior art base or the common general knowledge in the relevant art on or before the priority date of the claims herein.

SUMMARY OF THE INVENTION

In one aspect, the present invention provides a method of upgrading a software program in a computing device including the steps of:

transferring an upgrade control program into memory locations of the computing device to be upgraded, said upgrade control program containing instructions to control the transfer of data to the computing device;

applying a signal to a computer processing means in the computing device such that the processing means is caused to execute the instructions contained in said upgrade control program, said instructions causing the computing device to transfer data from a source device and write said data into memory locations of the computing device; and

writing at least a portion of said data into at least one said memory location.

Preferably, the memory locations in the computing device correspond to non-volatile memory locations such that once the upgrade control program is written to those memory locations a power failure to the computing device will not cause the loss of the instructions of the upgrade control program.

In a preferred embodiment, the source device from which data is transferred is a host device. The upgrade control program may be a relatively simple program requiring a relatively small number of instructions and preferably only manages communication with the host device for transferring data from that host device. Management of communication with a host device may include performing data integrity tests during the process of transferring data from a host device to the computing device.

In a particularly preferred embodiment, the data transferred from a host device is a "new" operating system for the computing device. Writing transferred data into the memory locations occupied by the instructions of the upgrade control program preferably occurs at the end of the data transfer process. In this instance, the upgrade control program transfers data into "available" memory locations until there are no further available memory locations at which point the upgrade control program directs the transfer of data into memory locations occupied by the upgrade control program instructions thereby causing the transferred data to overwrite the upgrade control program. This results in the optimal use of available memory and thus the entire memory of the computing device is available for transferred data such as a new operating system.

The upgrade control program preferably causes the computing device to be reset after the upgrade control program overwrites itself in memory thus causing the new operating system instructions to be executed by the computing device. At this stage, the computing device will then operate according to the upgraded operating system.

The transfer of data to the computing device is preferably monitored by the host device. In a preferred embodiment, the host device is a personal computer that is connected via a communication channel with the computing device to be upgraded. The communication channel between the host device and the computing device may be a Universal Serial Bus (USB), Blue Tooth ™ or an ethernet connection. The upgrade control program preferably co-operates with the program executing on the host device that controls the transfer of data to the device such that if an error occurs during the transfer of data a recovery process is initiated. Various problems may occur during a transfer of data from a host device to a computing device including a power failure to either the host device or the computing device, an inadvertent reset of the computing device or the host device or an inadvertent disconnection of the communication channel between the devices. Preferably, any recovery process initiated avoids the need for the data transfer process to recommence from the start.

In a preferred embodiment, the upgrade control program is stored on the host device and is transferred to the computing device as part of the first phase of an upgrade to the computing device operating system. The upgrade control program will generally be a much smaller program as compared with an operating system program for the computing device. Accordingly, the time required to transfer an upgrade control program to the computing device will be significantly less as compared with the time required to transfer an operating system program.

Of course, it is very difficult to recover from any problem occurring during the transfer of an upgrade control program to a computing device. However, as this transfer will require much less time as compared with transferring an upgraded operating system, the risk of an unrecoverable problem occurring is significantly reduced. Once a successful transfer of an upgrade control program from a host device to a computing device is completed, any problems occurring during a subsequent transfer of data to the computing device may be overcome.

In a preferred embodiment, all information relating to the progress of a data transfer is maintained on the host device. In this instance, if a problem were to occur with the computing device during a transfer then the host device can resume the data transfer when the problem with the computing device has been resolved. During a data transfer process, a host device is considered to be a more stable platform as it is not having its operating system altered. Accordingly, it is more likely that problems will occur with the computing device as compared with a host device and as such is it preferable that the host device retain the status information relating to the progress of a data transfer.

Whilst it is preferable that any recovery of a data transfer process avoid recommencing the process from the start, ultimately, if status information relating to the progress of a data transfer is lost or corrupted as a result of a problem during the process, the data transfer may be reinitiated from the beginning.

Prior to an upgrade process commencing, it is preferable that the host device tests whether the new operating system is compatible with the computing device. This may be performed by a comparison of identification numbers associated with the operating system and the computing device. Further, it is preferable that the host device initiate tests with respect to the supply of power to the computing device and to check whether compatible drivers are installed on the host device.

In another embodiment of the invention, the source device is a data storage means that is preferably non-volatile. This particular embodiment of the invention is well suited to upgrading the BIOS of a Personal Computer or other computing device that is connected to a relatively large non-volatile storage means.

In another aspect, the present invention provides an upgrade control software program including instructions to control the transfer of data to a computing device, the upgrade control program being transferred into memory locations in a computing device to be upgraded and upon execution of instructions contained in said upgrade control program, said instructions causing the computing device to transfer data from a source device and write said data into memory locations of the computing device said instructions further causing at least a portion of said data to be written into at least one said memory location.

In a further aspect, the present invention provides a computing device with upgraded software wherein said upgrade to the computing device software included the steps of:

transferring an upgrade control program into memory locations of the computing device to be upgraded, said upgrade control program containing instructions to control the transfer of data to the computing device;

applying a signal to a computer processing means in the computing device such that the processing means is caused to execute the instructions contained in said upgrade control program, said instructions causing the computing device to transfer data from a source device and write said data into memory locations of the computing device; and

writing at least a portion of said data into at least one said memory location.

BRIEF DESCRIPTION OF THE DRAWINGS

A preferred embodiment of the invention will now be described with reference to the following diagrams in which:

Figure 1 is a diagrammatic representation of the basic memory components of a computing device;

Figure 2 is a diagrammatic representation of the basic memory components of a computing device detailing the relative location of a "new" operating system image and an "old" operating system during the process of upgrading the operating system of the computing device;

Figure 3 is a process flow chart detailing the primary method steps of a method upgrading an operating system in a computing device according to an embodiment of the invention; and

Figures 4 to 10 are diagrammatic representations of particular method steps identified in Figure 3.

DETAILED DESCRIPTION OF A PREFERRED EMBODIMENT

With reference to Figure 3, a preferred embodiment of a method according to the present invention is detailed. In this particular embodiment, the upgrade control program is sometimes referred to as a "bootloader".

At the commencement of the method (step 50) a determination is made as to whether the upgrade process is commencing for the first time or the process had previously been commenced and is currently in a recovery phase. This determination (step 55) determines the next method step to be executed.

In the event that the method does not need to enter a recovery phase, the method proceeds to step 60 wherein a communication program is downloaded to the host device. The communication program enables the host device to communicate with the computing device that requires an operating system upgrade and upon establishing communication between the devices, the host device obtains an identification code of the computing device (step 65).

The method then proceeds to step 70 wherein a user operating the host device selects the upgraded operating system to be transferred to the computing device (step 70). Upon selecting an upgraded operating system for transfer to the computing device, the host device retrieves the previously obtained device information (step 75) and performs a comparison of the identification codes for the upgrade operating system and the computing device to determine whether they are compatible (step 80). If the operating system selected by the user is not compatible with the computing device that the user wishes to upgrade, the method reverts to step 70 wherein the user is prompted to select another operating system for transfer to the computing device (step 70).

When a compatible operating system for the computing device has been selected the method proceeds to step 85 wherein a back-up version of that part of the Operating System that will be overwritten by the upgrade control program is created. This step of the method is further illustrated in Figure 4.

At step 90, the host device transfers an upgrade control program to the computing device and stores the upgrade control program in the flash memory block of the computing device that was previously backed-up (step 85). This step of the method is further illustrated in Figure 5. A reset signal is then applied to the computing device such that the upgrade control program effectively takes control of the computing device.

Up to this stage, the computing device was controlled by execution of the old" operating system which necessarily includes the facility to transfer data from

a host device which enables the transfer of an upgrade control program in the form of the OS bootloader of the preferred embodiment.

Having reset the computing device and causing it to operate under the control of the bootloader program, the next step in the method is to create a backup copy of the contents of the RAM of the computing device (step 95). This step of the method is further illustrated in Figure 6. Upon completion of this step, a flag is set to indicate that this method step has been completed successfully (Set Backup Stage to 1). The contents of the RAM of the computing device are copied (ie backed-up) on the host device.

Having successfully completed the creation of a back-up copy of the contents of the RAM on the host device, the method proceeds to step 100 wherein the creation of a back-up copy of the "old" operating system on the host device occurs. This step of the method is further illustrated in Figure 7 wherein a back-up version of the remainder of the "old" operating system (that was not backed-up during step 85) is created. Upon successfully completing this step of the method, a flag is set to indicate same (Set Backup Stage to 2). The "old" operating system is copied to the host device such that it may be subsequently restored if the user decides that they prefer the "old" operating system as compared with the upgraded version.

The next step in the method (step 105) compresses and encrypts the back-up versions of the contents of the computing device RAM and the "old" operating system. Compressing the Operating System reduces the space required in the host device and encrypting the image of the operating system prevents copying of the operating system to computing devices that are not licensed to use the software. For example, a user may have more than one computing device that have a range of versions of operating systems. Encrypting operating system software during the creation of a back-up on a host device and including the unique identification number of a device in the encrypted back-up prevents a user from upgrading the operating system in other computing devices. In this respect, when attempting to restore an "old" operating system to a device, the restoration process should only allow restoration to the computing device licensed for that software.

The next step in the method (step 110) is the transfer of the data representing the new operating system image from the host device to the computing device. This step in the method also includes restoring the original contents of the RAM subsequent to the transfer of the operating system image into flash memory of the computing device and is further illustrated in Figure 8. Upon successful completion of this step in the method a flag is set to indicate same (Set Backup Stage to 3).

Having successfully transferred the new operating system to the flash memory of the computing device, the upgrade control program is overwritten (step 115) which is further illustrated in Figure 9.

At step 120, the method then determines whether the user requires the computing device RAM to be restored. If the user requests the RAM to be restored, the contents of the computing device RAM that was previously backed-up in the host device is transferred to the computing device RAM. This step is further illustrated in Figure 10.

At this stage, the method continues to step 140 wherein a reset signal is applied to the computing device. If the user elects not to restore the contents of the computing device RAM (refer step 120), the method proceeds directly to step 140. At this stage, the computing device operating system has been effectively upgraded and the upgrade process is considered to have been completed.

The process described to this point was based on the assumption that the upgrade process was started for the first time. Alternatively, if a problem occurs during the operating system upgrade process, the method of the present invention attempts to recover the upgrade process without requiring intervention by a user. In this instance at step 55, a determination as to whether the process was commencing for the first time or required a recovery operation would cause the method to proceed to step 150.

At this stage, the host device obtains from the computing device an identification code and also retrieves status information with respect to the upgrade process (step 155). Subsequently, the host device sets a recovery flag to a "true" state (step 160) and obtains recovery status information (step 165) indicating the last stage of the back-up process that was successfully completed. Depending upon the last back-up stage that was successfully completed, the

method will proceed to either step 95 (stage 1 successfully completed), step 100 (stage 2 successfully completed), step 110 (stage 3 successfully completed), or step 115 if the fourth stage of the back-up process had been successfully completed.

In another embodiment of the invention, the upgrade process is used to upgrade the BIOS of a Personal Computer. In this instance, an upgrade control program is written into the non-volatile memory where the BIOS resides and the Personal Computer is then reset. This causes the Personal Computer to act under the control of the upgrade control program which may then direct the transfer of a new BIOS from a non-volatile storage device connected to the Personal Computer such as a floppy disk or a Hard Disk Drive.

If an error were to occur during the upgrade process, resetting the Personal Computer would cause the upgrade control program to resume control of the upgrade and direct the process to continue from where it was interrupted.

In accordance with the invention, transferring data to the BIOS results in the new BIOS being written into memory locations occupied by the upgrade control program thus overwriting the upgrade control program.

CONCLUSION

The present invention embodies many advantages as compared with the usual approach to upgrading the operating system of a computing device such as an embedded system or a personal digital assistant. In particular, the method and system of the present invention provides users with an automated system that attempts to recover the upgrade process in the event that any errors or problems occur during the transfer of a new operating system image to the computing device.

Further, significant advantages result from storing an upgrade control program on a host device and downloading that program to the target computing device as and when an upgrade to an operating system is required. In particular, upgrades to the upgrade control program may be issued and used in combination with a program executed on a host device for the purpose of upgrading the operating system of a computing device. Also, as a result of the upgrade control program only being stored temporarily in the computing device for the purpose of

effecting an upgrade to the operating system of the computing device, the upgrade control program does not consume valuable memory resource in the computing device.

As compared with the usual process for upgrading software in a computing device, the present invention significantly reduces the risk of an error or problem occurring during an upgrade process that could render a computing device inoperable.

The claims defining the present invention are as follows:

1. A method of upgrading a software program in a computing device including the steps of:

transferring an upgrade control program into memory locations of the computing device to be upgraded, said upgrade control program containing instructions to control the transfer of data to the computing device;

applying a signal to a computer processing means in the computing device such that the processing means is caused to execute the instructions contained in said upgrade control program, said instructions causing the computing device to transfer data from a source device and write said data into memory locations of the computing device; and

writing at least a portion of said data into at least one said memory location.

2. A method according to claim 1 wherein the memory locations are non-volatile memory locations.

3. A method according to either claim 1 or claim 2 wherein said source device is a host computing device.

4. A method according to claim 1 wherein said source device is a non-volatile data storage device.

5. A method according to claim 3 wherein said upgrade control program performs data integrity tests as data is transferred from the host device to the computing device.

6. A method according to any one of the preceding claims wherein the upgrade control program causes the computing device to be reset upon completing the transfer of data to the computing device.

7. A method according to any one of claims 3 to 6 wherein said host device is a personal computer.

8. A method according to any one of claims 3 to 7 wherein said computing device is a personal digital assistant, the method effecting an upgrade to the operating system of the personal digital assistant.

9. A method according to any one of claims 3 to 8 wherein the host computing device maintains information relating to the progress of the transfer of data from the host device to the computing device such that, in the event of an error during a data transfer, the transfer may be re-commenced using the progress information maintained by the host computing device.

10. A method according to any one of claims 4 to 7 wherein said computing device is a personal computer and the non-volatile storage device is a Hard Disk Drive connected to the personal computer, the method effecting an upgrade to the BIOS of the personal computer.

11. An upgrade control software program including instructions to control the transfer of data to a computing device, the upgrade control program being transferred into memory locations in a computing device to be upgraded and upon execution of instructions contained in said upgrade control program, said instructions causing the computing device to transfer data from a source device and write said data into memory locations of the computing device said instructions further causing at least a portion of said data to be written into at least one said memory location.

12. A computing device with upgraded software wherein said upgrade to the computing device software included the steps of:

transferring an upgrade control program into memory locations of the computing device to be upgraded, said upgrade control program containing instructions to control the transfer of data to the computing device;

applying a signal to a computer processing means in the computing device such that the processing means is caused to execute the instructions contained in said upgrade control program, said instructions causing the computing device to transfer data from a source device and write said data into memory locations of the computing device; and

writing at least a portion of said data into at least one said memory location.