

SOFTWARE REFRESHED MEMORY DEVICE AND METHOD

Background of the Invention

Field of the Invention

[0001] This invention relates generally to computer memory devices and more particularly to memory devices that must be periodically refreshed to avoid losing data.

Description of the Related Art

[0002] Certain memory devices can maintain the information stored in the memory indefinitely, even when the power to the memory device is turned off. These memory devices are known as non-volatile memory devices. Some examples of non-volatile memory devices include magnetic random access memories (MRAMs), erasable programmable read only memories (EPROMs), and variations thereof.

[0003] Other memory devices require power to maintain the information stored in the memory. These memory devices, which are known as volatile memory devices, must be periodically refreshed to avoid losing data. One common example of a volatile memory device is a dynamic random access memory (DRAM), wherein voltages stored in capacitors represent digital bits of information. Because the voltage stored in a capacitor dissipates over time, the capacitors of a DRAM must be periodically re-charged to maintain the information stored in the DRAM.

[0004] Conventional volatile memory devices include a plurality of individual memory cells configured in an array, which typically comprises a configuration of intersecting rows and columns. To maintain the information stored in a conventional volatile memory device, each memory cell in the array is typically refreshed at least several times per second. For example, in some conventional DRAMs, each memory cell must be refreshed about once every 64 milliseconds to avoid losing the information stored in the DRAM. Accordingly, conventional volatile memory devices typically include refresh circuitry to ensure that every memory cell is refreshed at least as often as necessary to avoid losing data, which is commonly at least several times per second.

Summary of the Invention

[0005] In one embodiment, a method of refreshing a cell in a memory device comprises determining whether said cell is in a written state or in an erased state, and, if said cell is in said written state, refreshing said cell by issuing a refresh write instruction to said cell.

[0006] In another embodiment, a method of preserving data stored in a volatile memory device having a plurality of cells comprises, for each of said plurality of cells, determining whether said cell needs to be refreshed, and, if said cell needs to be refreshed, refreshing said cell.

[0007] In another embodiment, a method of preserving data stored in a volatile memory device having a plurality of cells comprises refreshing said plurality of cells, and waiting for a predetermined period of time lasting for at least about one second, wherein none of said plurality of cells is refreshed during said predetermined period of time.

[0008] In another embodiment, a method of preserving data stored in a volatile memory device having a plurality of memory cells comprises addressing a first memory cell, waiting for a first period of time since said first memory cell was addressed, and determining whether said first memory cell needs to be refreshed. If said first memory cell needs to be refreshed, the method further comprises determining whether system resources are available to refresh said first memory cell, and, if said first memory cell needs to be refreshed and if said system resources are not available, monitoring whether said system resources become available to refresh said first memory cell within a second period of time since said first memory cell was addressed. If said first memory cell needs to be refreshed and if said system resources do not become available within said second period of time, the method further comprises forcing said resources to be relinquished, such that said resources become available to refresh said first memory cell, and, if said first memory cell needs to be refreshed, refreshing said first memory cell using said available system resources. The method further comprises addressing a second memory cell.

[0009] In another embodiment, a method of avoiding loss of data in a volatile memory device comprises establishing a deadline by which said volatile memory device must be refreshed, and monitoring whether resources are available to refresh said volatile

memory device. If resources do not become available to refresh said volatile memory device within a first predetermined time period before said deadline, the method further comprises forcing said resources to be relinquished, such that said resources become available to refresh said volatile memory device. The method further comprises using said available resources to refresh said volatile memory device before said deadline.

[0010] In another embodiment, a computer system comprises a processor, a memory device coupled to said processor, wherein said memory device comprises a plurality of cells that must be periodically refreshed, and a software module that, when executed by said processor, refreshes said plurality of cells.

[0011] In another embodiment, a memory device comprises a plurality of cells that must be periodically refreshed, wherein said plurality of cells are configured to be refreshed in response to a write instruction or an erase instruction received from a processor, and wherein said memory device is not configured to generate or to receive a refresh control signal that differs from said write instruction or from said erase instruction.

[0012] In another embodiment, a memory device comprises a plurality of cells that must be periodically refreshed, wherein said plurality of cells are configured to be refreshed in response to a write instruction or an erase instruction received from a processor, and wherein said memory device is not configured to generate or to receive a refresh control signal that differs from said write instruction or from said erase instruction. Each of said plurality of cells comprises a programmable metallization cell, which comprises a cell body having a top surface, wherein said cell body comprises a chalcogenide-metal ion glass and two electrodes disposed at said top surface, wherein said electrodes are spaced a distance apart from one another.

[0013] In another embodiment, a memory device comprises a plurality of cells that must be periodically refreshed, wherein said plurality of cells are configured to be refreshed in response to a write instruction or an erase instruction received from a processor.

Brief Description of the Drawings

[0014] Figure 1 illustrates a computer system that includes a volatile memory device.

[0015] Figure 2 illustrates a computer system having a software refreshed memory device in accordance with one embodiment of the present invention.

[0016] Figure 3 illustrates a method of refreshing a memory cell in accordance with one embodiment of the present invention.

[0017] Figure 4 illustrates a method of refreshing a plurality of memory cells in accordance with one embodiment of the present invention.

[0018] Figure 5 illustrates a method of refreshing a plurality of memory cells in accordance with one embodiment of the present invention.

Detailed Description

[0019] Figure 1 illustrates a computer system 100 that includes a conventional volatile memory device 110. The computer system 100 also comprises a processor 120 coupled to the memory device 110 via a bus 130. The memory device 110 comprises an address/data/control module 140 and memory refresh circuitry 150, both of which are coupled to the processor 120 via the bus 130. The memory device 110 also comprises a memory array 160 coupled to the address/data/control module 140 via line 200 and to the memory refresh circuitry 150 via line 210. Those of ordinary skill in the art will understand that lines 200 and 210 may be implemented in the memory device 110 as part of a single physical bus. The memory array 160 comprises a plurality of memory cells 170, which are interconnected by a plurality of intersecting rows 180 and columns 190.

[0020] To perform operations in the memory device 110, the processor 120 transmits certain signals to the memory device 110 via the bus 130. For example, to read data stored at a particular memory address, the processor 120 issues a read command, together with the memory address, to the bus 130. The address/data/control module 140 receives and processes the read command by accessing the memory array 160 via line 200. Specifically, the address/data/control module 140 generates a read control signal on line 200 and addresses the desired memory cell 170 by activating the appropriate row 180 and column 190. The address/data/control module 140 then receives the data stored at the addressed memory cell 170 via line 200, and passes the data to the processor 120 via the bus 130.

[0021] In addition, to write data to a particular memory address, the processor 120 issues a write command, together with the memory address and the data to be stored, to the

bus 130. The address/data/control module 140 receives and processes the write command by generating a write control signal on line 200 and addressing the desired memory cell 170, as described above. The address/data/control module 140 then passes the data to be stored to the addressed memory cell 170 via line 200.

[0022] Because the data stored in the memory cells 170 dissipates over time, the memory cells 170 must be periodically refreshed to avoid losing the data stored in the memory device 110. The primary function of the memory refresh circuitry 150 is to perform these periodic memory refresh operations. When the memory refresh circuitry 150 determines that it is necessary to perform a memory refresh operation, the memory device 110 is made unavailable to the processor 120 to perform other operations, such as a read operation or a write operation.

[0023] To perform a memory refresh operation, the memory refresh circuitry 150 typically sequentially addresses the rows 180 of the memory array 160. When a given row 180 of the array 160 is addressed, the memory refresh circuitry generates a refresh control signal on line 210, which causes all of the memory cells 170 in the addressed row 180 to be refreshed simultaneously. By refreshing a large number of memory cells 170 simultaneously, the memory refresh circuitry 150 advantageously reduces the amount of time required to perform a memory refresh operation, thereby reducing the amount of time that the memory device 110 is unavailable to the processor 120 to perform other operations.

[0024] Because the memory device 110 is unavailable to perform other operations during a memory refresh operation, it is desirable to time memory refresh operations such that they occur at times when they will not conflict with requests from the processor 120. Therefore, the memory refresh circuitry 150 is often configured to identify time periods when the processor 120 is not likely to issue requests to the memory device 110, and to perform memory refresh operations during these identified time periods. On the other hand, because power is required to perform a memory refresh operation, it is desirable to minimize the number of memory refresh operations performed. Thus, the memory refresh circuitry 150 is also often configured to perform memory refresh operations as infrequently as possible, while preserving the data stored in the memory device 110.

[0025] Nevertheless, to avoid losing data stored in conventional volatile memory devices 110, the memory refresh circuitry 150 must perform memory refresh operations frequently, often many times per second. For example, in some embodiments, each memory cell 170 must be refreshed at least about once every 64 milliseconds to avoid losing data. In these embodiments, to avoid losing the information stored in the memory device 110, the memory refresh circuitry 150 must ensure that every memory cell 170 is refreshed at least more often than about 15 times per second.

[0026] Figure 2 illustrates a computer system 250 having a software refreshed memory device 260 in accordance with one embodiment of the present invention. The computer system 250 also comprises a processor 270 coupled to the memory device 260 via a bus 280. The computer system 250 further comprises a set of memory refresh instructions 290, which are implemented in software that can be executed by the processor 270. The memory device 260 comprises an address/data/control module 300 which is coupled to the processor 270 via the bus 280. The memory device 260 also comprises a memory array 310 coupled to the address/data/control module 300 via line 350. The memory array 310 comprises a plurality of memory cells 320, which are interconnected by a plurality of intersecting rows 330 and columns 340.

[0027] The computer system 250 illustrated in Figure 2 can perform memory read and write operations using the same methods described above in connection with Figure 1. In addition to these methods, however, the computer system 250 can perform memory read and write operations in a variety of other ways that are well-known to those of ordinary skill in the art.

[0028] In some embodiments, the memory cells 320 of the software refreshed memory device 260 comprise volatile memory cells, which are preferably more stable than conventional DRAM memory cells. For example, in one embodiment, the memory cells 320 can avoid losing data even though the time interval between successive memory refresh operations is about 0.1 seconds. In another embodiment, the time interval between successive memory refresh operations can be about one second. In yet another embodiment, the time interval between successive memory refresh operations can be about one hour. In

yet another embodiment, the time interval between successive memory refresh operations can be about one day to one week.

[0029] In some embodiments, the memory cells 320 of the software refreshed memory device 260 comprise programmable conductor random access memory (PCRAM) cells, which are described in U.S. Patent Nos. 5,761,115, 5,896,312, 5,914,893, 6,084,796 to Kozicki et al. ("the Kozicki patents"), in U.S. Patent No. 6,348,365 to Moore et al. ("the Moore patent"), and in the following co-pending U.S. patent applications: Serial No. 10/121,792 entitled "Method of Manufacture of Programmable Conductor Memory" filed April 10, 2002, Serial No. 10/121,790 entitled "Programmable Conductor Memory Cell Structure and Method Therefor" filed April 10, 2002, and Serial No. 10/121,794 entitled "Thin Film Diode Integrated with Chalcogenide Memory Cell" filed April 10, 2002. The Kozicki patents, the Moore patent, and these co-pending patent applications are hereby incorporated in their entireties by this reference. As discussed in more detail in these references, a PCRAM cell comprises a pair of electrodes and can exist in one of two possible states. In the first state, an electrical short exists between the electrodes of the PCRAM cell. In the second state, an open circuit exists between the electrodes of the PCRAM cell.

[0030] Some embodiments of a PCRAM cell comprise a glass ion conductor, such as a chalcogenide-metal ion glass, and two electrodes disposed at the surface of the glass ion conductor and spaced a distance apart from one another. In one embodiment, a PCRAM cell comprises germanium selenide with a Group IV metal (e.g., silver) dissolved therein, such as Ag/Ge₃Se₇. Preferably, one of the electrodes comprises a Group IV metal, and the glass element of the PCRAM cell contains the same metal.

[0031] In operation, when a voltage having a first polarity is applied across the electrodes of a PCRAM cell, a conductive path is created between the electrodes along the sidewalls of the via in which the glass element is formed. When a voltage having the opposite polarity is applied across the electrodes, the metal ions re-dissolve into the cell body, thereby causing the conductive path to disappear. The presence or absence of a conductive path within a PCRAM cell can be detected by measuring the electrical resistance between the electrodes. When a conductive path is present, an electrical short exists between the electrodes, and the resistance between the electrodes is low (e.g., on the order of

milliohms). On the other hand, when no conductive path is present, an open circuit exists between the electrodes, and the resistance between the electrodes is high (*e.g.*, on the order of megaohms).

[0032] The features of Figure 2 are referenced throughout the discussion below of operation processes.

[0033] Typically, the memory cells 320 of the memory device 260 are capable of existing in one of two states, *i.e.*, a "written" state or an "erased" state. For example, if a memory cell 320 comprises a capacitor capable of holding a charge, the presence of a charge in the capacitor would correspond to the written state, and the absence of a charge in the capacitor would correspond to the erased state. Similarly, if a PCRAM cell acts as a memory cell 320, the presence of a conductive path between the electrodes would correspond to the written state, while the absence of a conductive path between the electrodes would correspond to the erased state. Those of ordinary skill in the art will understand that, in general, the presence of an element of interest within a memory cell 320 will correspond to the written state, whereas the absence of an element of interest will correspond to the erased state.

[0034] As discussed above in connection with Figure 1, when a memory cell 170 of a conventional volatile memory device 110 is placed in a particular state, the memory cell 170 remains in the given state for a relatively short period of time, such as, for example, about 64 milliseconds. Because such conventional memory cells 170 retain their assigned states for such a short time period, each memory cell 170 must be refreshed often, such as, for example, at least more often than about 15 times per second.

[0035] By contrast, when a memory cell 320 of the software refreshed memory device 260 illustrated in Figure 2 is placed in a particular state, the memory cell 320 advantageously remains in the given state for a relatively long period of time. For example, in some embodiments, each memory cell 320 can maintain a given state for a period of seconds, minutes, hours, days, weeks, or longer. Accordingly, the data stored in the memory device 260 can be preserved while performing memory refresh operations less frequently, such as, for example, about once every few weeks, rather than several times per second.

[0036] Because memory refresh operations can occur less frequently, the computer system 250 can advantageously perform these memory refresh operations by executing a set of memory refresh instructions 290 implemented in software, rather than in hardware. For example, in some embodiments, the memory refresh instructions 290 constitute part of the operating system of the computer system 250.

[0037] By implementing the memory refresh instructions 290 in software, rather than in hardware, the memory device 260 can advantageously be simplified. For example, the need for the memory refresh circuitry 150 and for a unique refresh control signal on line 210, as illustrated in Figure 1, are advantageously eliminated. Another advantage of implementing the memory refresh instructions 290 in software, rather than in hardware, is that the processor 270 (Figure 2) can perform more sophisticated algorithms, as compared to the memory refresh circuitry 150 (Figure 1), for determining when to perform a memory refresh operation.

[0038] Figure 3 illustrates a method of refreshing a memory cell 320 (Figure 2) in accordance with one embodiment of the present invention. In a first step 400, the process begins. In a next step 402, the processor 270 reads the state of the memory cell 320. The state of the memory cell 320 is assigned a logical value, which corresponds to a digital bit of data. For example, in some embodiments, the written state of a memory cell 320 may correspond to a logical "1", while the erased state of the memory cell 320 corresponds to a logical "0". In some alternative embodiments, on the other hand, the written state of a memory cell 320 may correspond to a logical "0", while the erased state of the memory cell 320 corresponds to a logical "1".

[0039] In a step 404, the processor 270 determines whether the data bit stored in the memory cell 320 is a "1" or a "0". If the data bit is a "1", then, in a step 406, the processor 270 writes a "1" to the memory cell 320. For example, if a "1" corresponds to the written state, then, during step 406, the processor 270 issues a "write instruction" to the memory cell, *i.e.*, a write command is issued, and the memory cell 320 is placed in the written state. If, during step 404, the processor 270 determines that the data bit stored in the memory cell 320 is a "0", then, in a step 408, the processor 270 writes a "0" to the memory cell 320. For example, if a "0" corresponds to the erased state, then, during step 408, the

processor 270 issues an "erase instruction" to the memory cell, *i.e.*, a write command is issued, and the memory cell 320 is placed in the erased state.

[0040] A write instruction issued during a memory refresh operation can be the same as a write instruction issued during a standard write operation to the memory device 260. Similarly, an erase instruction issued during a memory refresh operation can be the same as an erase instruction issued during a standard write operation to the memory device 260. Accordingly, as discussed above, the need for a unique refresh control signal, which is utilized only during memory refresh operations, is advantageously eliminated. After the processor 270 completes step 406 or step 408, then, in a final step 410, the process ends.

[0041] In the method illustrated in Figure 3, it is assumed that a memory cell 320 (Figure 2) will not remain in the written state or in the erased state indefinitely unless it is periodically refreshed by issuing a write instruction or an erase instruction, whichever is appropriate, to the memory cell 320. In some embodiments, however, the memory cells 320 tend to revert to one particular state over time. For example, in some embodiments, the memory cells 320 tend to revert to the erased state over time. In these embodiments, when a memory cell 320 is placed in the written state, the memory cell 320 will not remain in the written state indefinitely unless it is periodically refreshed by performing a write operation on the memory cell 320. On the other hand, because the natural tendency of the memory cells 320 is to revert to the erased state, a memory cell 320 placed in the erased state will remain in this state indefinitely, without needing to be refreshed. In these embodiments, the method illustrated in Figure 3 can advantageously be simplified by eliminating either step 406 or step 408, whichever corresponds to the erased state.

[0042] Moreover, in the method illustrated in Figure 3, it is assumed that each memory cell 320 (Figure 2) is capable of existing in one of only two states. Nevertheless, in light of the present disclosure, those of ordinary skill in the art will understand how the method illustrated in Figure 3 can be modified if the memory cells 320 are capable of existing in more than two states.

[0043] If a memory refresh operation comprises the method illustrated in Figure 3, then each memory cell 320 (Figure 2) is refreshed individually, rather than refreshing a large number of memory cells 320 simultaneously, as discussed above in connection with

Figure 1. By evaluating each memory cell 320 individually, the processor 270 can determine whether each individual memory cell 320 needs to be refreshed, thereby advantageously avoiding performing unnecessary refresh operations on memory cells 320 that do not need to be refreshed.

[0044] Figure 4 illustrates a method of refreshing a plurality of memory cells 320 (Figure 2) in accordance with one embodiment of the present invention. To perform this method, the computer system 250 maintains a counter having a value which corresponds to a particular address in the memory array 190. In a first step 450, the memory cell 320 at the address corresponding to the current value of the counter is refreshed, using the method illustrated in Figure 3 or another suitable method. In a next step 452, the value of the counter is incremented, and in a next step 454, a timer is reset and started.

[0045] In a step 456, the processor 270 determines whether the timer has exceeded a predetermined minimum wait time. The appropriate value for the minimum wait time can be determined by considering a number of factors, such as, for example, the maximum time that a memory cell 320 can retain its assigned state, the time required to refresh a memory cell 320, the number of memory cells 320 to be refreshed, and the like. As discussed above, this predetermined minimum wait time can advantageously be a relatively long period of time, such as, for example, a period of seconds, minutes, hours, days, or longer. In one embodiment, the predetermined minimum wait time is a period of about one minute. In another embodiment, the minimum wait time is a period of about one hour. In yet another embodiment, the minimum wait time is a period of about one day to one week.

[0046] If the minimum wait time has not yet been reached, then, in a step 458, the processor 270 determines whether the memory cell 320 at the address corresponding to the current value of the counter needs to be refreshed. A number of different conditions may indicate that the memory cell 320 at the current address does not need to be refreshed. For example, as discussed above, in some embodiments, when a memory cell 320 is in the erased state, the memory cell 320 does not need to be refreshed. Moreover, if the processor 270 performs a write operation to the memory cell 320 sometime after the timer is reset and started during step 454, then the memory cell 320 does not need to be refreshed until the next memory refresh cycle.

[0047] If, while waiting for the timer to reach the minimum wait time, the processor 270 determines that the memory cell 320 at the current address does not need to be refreshed, then the process returns to step 452, where the value of the counter corresponding to the current memory address is incremented, and the process continues, as described above. On the other hand, if, once the timer reaches the minimum wait time, the memory cell 320 at the current address still needs to be refreshed, then the process proceeds to a step 460, where the processor 270 determines whether system resources are available to refresh the memory cell 320. In making this determination, the processor 270 may evaluate a wide variety of factors, such as, for example, the demands on the processor 270 and on the memory device 260 or other devices in the computer system 250, and the like.

[0048] If system resources are available, then the process returns to step 450, where the memory cell 320 at the current address is refreshed, and the process continues, as described above. On the other hand, if system resources are not available to refresh the memory cell 320, then, in a step 462, the processor 270 determines whether the timer has exceeded a predetermined maximum wait time. As with the minimum wait time, the appropriate value for the maximum wait time can be determined by considering a number of factors, such as, for example, the maximum time that a memory cell 320 can retain its assigned state, the time required to refresh a memory cell 320, the number of memory cells 320 to be refreshed, and the like. As discussed above, this predetermined maximum wait time can advantageously be a relatively long period of time, such as, for example, a period of seconds, minutes, hours, days, weeks, or longer. In one embodiment, the maximum wait time is a period of about one hour. In another embodiment, the maximum wait time is a period of about one week. In yet another embodiment, the maximum wait time is a period of about one month.

[0049] If the maximum wait time has not yet been reached, then, in a step 464, the processor 270 determines whether the memory cell 320 at the address corresponding to the current value of the counter needs to be refreshed. As discussed above in connection with step 458, a number of different conditions may indicate that the memory cell 320 at the current address does not need to be refreshed.

[0050] If the memory cell 320 at the current address does not need to be refreshed, then the process returns to step 452, where the value of the counter corresponding to the current memory address is incremented, and the process continues, as described above. On the other hand, if the memory cell 320 at the current address needs to be refreshed, then the processor 270 continues to monitor whether system resources have become available to refresh the memory cell 320.

[0051] Once the timer reaches the predetermined maximum wait time, if system resources have not become available and the memory cell 320 at the current address still needs to be refreshed, then the process proceeds to a step 466, where the processor 270 forces certain system resources to be relinquished by other processes, such that the necessary resources become available to refresh the memory cell 320. The process then returns to step 450, where the memory cell 320 at the current address is refreshed, and the process continues, as described above.

[0052] In one embodiment, the process illustrated in Figure 4 is repeatedly performed, without interruption, by the processor 270. In this embodiment, the memory refresh operation is an ongoing process, which is constantly occurring in the background of other processes being executed by the processor 270. The minimum wait time and the maximum wait time can advantageously be selected and adjusted such that each memory cell 320 is refreshed as infrequently as possible, while preserving the information stored in the memory device 310.

[0053] Figure 5 illustrates another method of refreshing a plurality of memory cells 320 in accordance with one embodiment of the present invention. In a first step 500, the process begins, and in a next step 502, a timer is reset and started. As with the method illustrated in Figure 4, the computer system 250 of Figure 2 maintains a counter having a value which corresponds to a particular address in the memory array 310 to perform the method illustrated in Figure 5. In a step 504, the memory cell 320 at the address corresponding to the current value of the counter is refreshed, using the method illustrated in Figure 3 or another suitable method. In a next step 506, the value of the counter is incremented.

[0054] In a step 508, the processor 270 determines whether the memory refresh operation is complete. A number of conditions may indicate that the memory refresh operation is complete. For example, the memory refresh operation may be considered complete when each memory cell 320 in a memory array 310 has been refreshed, or when each memory cell 320 in a particular block of memory has been refreshed.

[0055] If the memory refresh operation is not yet complete, then, in a step 510, the processor 270 determines whether system resources are available to refresh the memory cell 320 at the address corresponding to the current value of the counter. If system resources are available, then the process returns to step 504, where the memory cell 320 at the current address is refreshed, and the process continues, as described above.

[0056] On the other hand, if system resources are not available to refresh the memory cell 320, then, in a step 512, the processor 270 determines whether the timer has exceeded a predetermined maximum wait time. As discussed above in connection with Figure 4, the appropriate value for the maximum wait time can be determined by considering a number of factors, such as, for example, the maximum time that a memory cell 320 can retain its assigned state, the time required to refresh a memory cell 320, the number of memory cells 320 to be refreshed, and the like. Moreover, as discussed above, this predetermined maximum wait time can advantageously be a relatively long period of time, such as, for example, a period of seconds, minutes, hours, days, weeks, or longer.

[0057] In one embodiment, the maximum wait time discussed above in connection with Figure 4 corresponds to the maximum time that the processor 270 can wait before it forces system resources to become available to refresh an individual memory cell 320. In the process illustrated in Figure 5, on the other hand, the maximum wait time corresponds to the maximum time that the processor 270 can wait before it forces system resources to become available to refresh all of the remaining memory cells 320 in the memory array 310 or in the block of memory being refreshed. Thus, in one embodiment, the maximum wait time is a period of about one day. In another embodiment, the maximum wait time is a period of about one week. In yet another embodiment, the maximum wait time is a period of about one month to two months.

[0058] If the maximum wait time has not yet been reached, then process returns to step 510, where, as described above, the processor 270 continues to monitor whether system resources have become available to refresh the memory cell 320. Once the timer reaches the predetermined maximum wait time, if system resources have not become available, then the process proceeds to a step 514, where the processor 270 forces certain system resources to be relinquished by other processes, such that the necessary resources become available to refresh the memory cell 320. The process then returns to step 504, where the memory cell 320 at the current address is refreshed, and the process continues, as described above.

[0059] This process repeats until, during step 508, the processor 270 determines that the memory refresh operation is complete. Once this determination is made, the process then proceeds to a step 516, where the memory address counter is reset. In a final step 518, the process ends.

[0060] In one embodiment, the process illustrated in Figure 5 is repeated by the processor 270 periodically. In this embodiment, the memory refresh operation is performed periodically, rather than being a constantly ongoing process, as described above in connection with Figure 4. The processor 270 can advantageously call and perform this process at regular time intervals or at irregular time intervals, depending upon the requirements of the computer system 250 and upon the longevity of the data stored in the memory cells 320.

[0061] The processes illustrated in Figures 3, 4 and 5 are merely examples of algorithms that can be implemented in the memory refresh instructions 290. Those of ordinary skill in the art will understand that these exemplary algorithms can be easily modified by adding, removing, or varying certain steps. Moreover, in light of the present disclosure, those of ordinary skill in the art will understand how to develop a wide variety of alternative algorithms.

I CLAIM:

1. A method of refreshing a cell in a memory device, comprising:
determining whether said cell is in a written state or in an erased state; and
if said cell is in said written state, refreshing said cell by issuing a refresh write instruction to said cell.
2. The method of Claim 1, further comprising, if said cell is in said erased state, refreshing said cell by issuing a refresh erase instruction to said cell.
3. A method of preserving data stored in a volatile memory device having a plurality of cells, comprising:
for each of said plurality of cells, determining whether said cell needs to be refreshed; and
if said cell needs to be refreshed, refreshing said cell.
4. A method of preserving data stored in a volatile memory device having a plurality of cells, comprising:
refreshing said plurality of cells; and
waiting for a predetermined period of time lasting for at least about 64 milliseconds, wherein none of said plurality of cells is refreshed during said predetermined period of time.
5. The method of Claim 4, wherein said predetermined period of time lasts for at least about one second.
6. The method of Claim 4, wherein said predetermined period of time lasts for at least about one week.
7. A method of preserving data stored in a volatile memory device having a plurality of memory cells, comprising:
addressing a first memory cell;
waiting for a first period of time since said first memory cell was addressed;
determining whether said first memory cell needs to be refreshed;
if said first memory cell needs to be refreshed, determining whether system resources are available to refresh said first memory cell;

if said first memory cell needs to be refreshed and if said system resources are not available, monitoring whether said system resources become available to refresh said first memory cell within a second period of time since said first memory cell was addressed;

if said first memory cell needs to be refreshed and if said system resources do not become available within said second period of time, forcing said resources to be relinquished, such that said resources become available to refresh said first memory cell;

if said first memory cell needs to be refreshed, refreshing said first memory cell using said available system resources; and
addressing a second memory cell.

8. The method of Claim 6, wherein said first period of time is within the range of about one second to about one week.

9. The method of Claim 6, wherein said second period of time is within the range of about one week to about two months.

10. A method of avoiding loss of data in a volatile memory device, comprising:
establishing a deadline by which said volatile memory device must be refreshed;

monitoring whether resources are available to refresh said volatile memory device;

if resources do not become available to refresh said volatile memory device within a first predetermined time period before said deadline, forcing said resources to be relinquished, such that said resources become available to refresh said volatile memory device; and

using said available resources to refresh said volatile memory device before said deadline.

11. A computer system comprising:

a processor;

a memory device coupled to said processor, wherein said memory device comprises a plurality of cells that must be periodically refreshed; and

a software module that, when executed by said processor, refreshes said plurality of cells.

12. The computer system of Claim 11, further comprising an operating system configured to be executed by said processor, wherein said operating system comprises said software module.

13. The computer system of Claim 11, wherein said memory device comprises a programmable conductor random access memory.

14. The computer system of Claim 11, wherein each of said plurality of cells comprises a programmable metallization cell.

15. The computer system of Claim 14, wherein said programmable metallization cell comprises:

a cell body having a top surface, wherein said cell body comprises a chalcogenide-metal ion glass; and

two electrodes disposed at said top surface, wherein said electrodes are spaced a distance apart from one another.

16. The computer system of Claim 15, wherein said chalcogenide-metal ion glass comprises germanium selenide having a Group IV metal dissolved therein.

17. The computer system of Claim 16, wherein said chalcogenide-metal ion glass comprises $\text{Ag/Ge}_3\text{Se}_7$.

18. A memory device comprising a plurality of cells that must be periodically refreshed, wherein said plurality of cells are configured to be refreshed in response to a write instruction or an erase instruction received from a processor, and wherein said memory device is not configured to generate or to receive a refresh control signal that differs from said write instruction or from said erase instruction.

19. The memory device of Claim 18, wherein said memory device comprises a programmable conductor random access memory.

20. The memory device of Claim 18, wherein each of said plurality of cells comprises a programmable metallization cell.

21. The memory device of Claim 20, wherein said programmable metallization cell comprises:

a cell body having a top surface, wherein said cell body comprises a chalcogenide-metal ion glass; and

two electrodes disposed at said top surface, wherein said electrodes are spaced a distance apart from one another.

22. The memory device of Claim 21, wherein said chalcogenide-metal ion glass comprises germanium selenide having a Group IV metal dissolved therein.

23. The memory device of Claim 22, wherein said chalcogenide-metal ion glass comprises Ag/Ge₃Se₇.