(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
4 August 2016 (04.08.2016)

WIPO | PCT

(10) International Publication Number
WO 2016/122471 A1

(51) International Patent Classification:
G06F 12/16 (2006.01) *G06F 11/14* (2006.01)

(21) International Application Number:
PCT/US2015/013213

(22) International Filing Date:
28 January 2015 (28.01.2015)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **HEWLETT PACKARD ENTERPRISE DEVELOPMENT LP** [US/US]; 11445 Compaq Center Drive West, Houston, TX 77070 (US).

(72) Inventors: **PLANK, Jeffrey A.**; 11445 Compaq Center Dr. W., Houston, Texas 77070 (US). **HEINRICH, David F.**; 11445 Compaq Center Dr. W., Houston, Texas 77070 (US). **WANG, Han**; 11445 Compaq Center Dr. W., Houston, Texas 77070 (US). **RAYMOND, Patrick A.**; 11445 Compaq Center Dr. W., Houston, Texas 77070 (US). **VENUGOPAL, Raghavan V.**; 11445 Compaq Center Dr. W., Houston, Texas 77070 (US). **OLAWSKY, Barry L.**; 11445 Compaq Center Dr. W., Houston, Texas 77070 (US).

(74) Agents: **WARD, Aaron S.** et al.; Hewlett Packard Enterprise, 3404 E. Harmony Road, Mail Stop 79, Fort Collins, CO 80528 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

(54) Title: MEMORY MODULE PERSISTENT DATA BACK-UPS

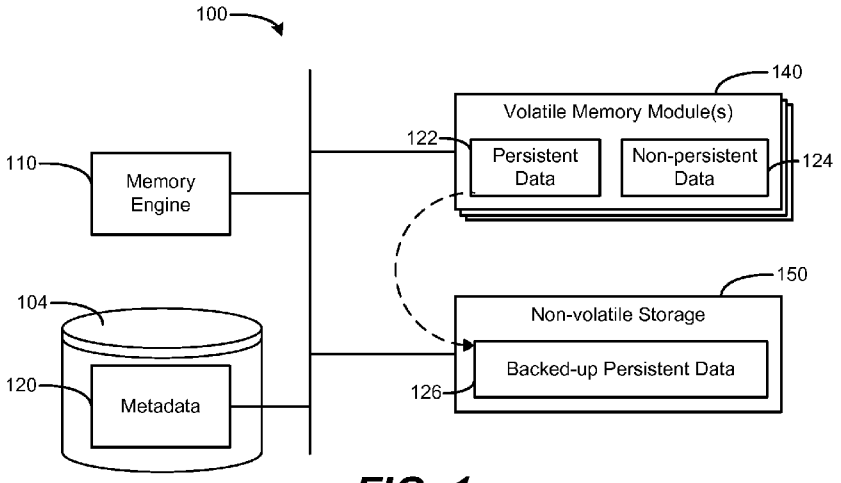

FIG. 1

(57) Abstract: An example device in accordance with an aspect of the present disclosure is to identify, according to metadata, persistent data to be backed up from a plurality of volatile memory modules. The persistent data is copied from the plurality of volatile memory modules to the non-volatile storage, while excluding the non-persistent data.

WO 2016/122471 A1

MEMORY MODULE PERSISTENT DATA BACK-UPS

BACKGROUND

[0001] Persistent memory solutions have been limited to a granularity of entire memory modules, and may need contiguous memory ranges, associated with the overhead of needing more expensive non-volatile memory. Supporting non-volatile memory can impose high costs and impose design constraints on applications that access the memory.

BRIEF DESCRIPTION OF THE DRAWINGS/FIGURES

[0002] FIG. 1 is a block diagram of a computing system including a memory engine and metadata according to an example.

[0003] FIG. 2 is a block diagram of a computing system including memory controller instructions, DMA engine instructions, and metadata instructions according to an example.

[0004] FIG. 3 is a block diagram of a computing system including a memory controller, a DMA engine, and a metadata engine according to an example.

[0005] FIG. 4 is a flow chart based on copying persistent data according to an example.

DETAILED DESCRIPTION

[0006] A given server may support large numbers of memory modules, and include a basic input/output system (BIOS) that may group volatile and non-volatile memory into different segments. However, applications to be run on the server may be allocated a single segment (linear region of memory) by the system, despite a desire to provide back-up memory support to the application.

Accordingly, a mismatch may exist between how the system allocates memory in different segments, in contrast to how applications are allocated to a single linear region of memory, creating difficulties when desiring to have an application interact with volatile and non-volatile memory.

[0007] Examples described herein may avoid such difficulties, by enabling support of persistent and non-persistent application data in memory, in a manner that is compatible with how applications may expect data to be presented. Allowing persistent data to coexist with non-persistent data in a single contiguous memory block as presented by the system (e.g., as presented to the operating system) can simplify the task of memory allocation for applications, especially applications that interact with non-volatile memory. Examples described herein may identify that a portion, rather than all, information stored in memory is to be retained after a power loss condition at the server. Such portion of the stored information, referred to herein as persistent data, may be distributed in many small blocks throughout the memory (e.g., throughout a plurality of memory modules). Accordingly, examples may avoid a need for all memory modules to be non-volatile. In contrast, a subset of memory modules may be provided as non-volatile memory (i.e., memory that is relatively more expensive), with the remainder of the memory modules being volatile (e.g., less expensive commodity memory modules).

[0008] FIG. 1 is a block diagram of a computing system 100 including a memory engine 110 and metadata 120 according to an example. The metadata 120 may be stored in storage 104. The memory engine 110 and metadata 120 may interact with volatile memory module(s) 140 and non-volatile storage 150. A volatile memory module 140 includes persistent data 122 and non-persistent data 124. The non-volatile storage 150 includes backed-up persistent data 126, copied from persistent data 122 of the volatile memory modules 140.

[0009] The memory engine 110 is to generally coordinate data access/flow in the memory (volatile memory modules 140 and non-volatile storage 150), including identifying locations of the persistent data 122 and non-persistent data 124 throughout a plurality of volatile memory modules 140. Thus, the memory engine 110 may assist in building the metadata 120, such as by identifying

characteristics (e.g., address/location) of persistent data 122 to be stored as metadata 120. In some examples, the memory engine 110 may include a memory controller and/or a DMA engine, and functionality described in terms of the memory engine 110 may be provided by a memory controller and/or a DMA engine.

[0010] The memory engine 110 may allow access to the memory 140, 150 directly, e.g., independent of a central processing unit (CPU) of the computing system 100 (not shown in FIG. 1). The memory engine 110 may serve as a data mover, and may copy/move the persistent data 122 from the volatile memory module 140 to the non-volatile storage 150 (which may include a memory module(s)), and vice-versa. In some examples, the memory engine 110 may be a memory controller and/or a direct memory access (DMA) engine.

[0011] The memory, such as volatile memory modules 140 and/or non-volatile storage 150, may be formed of memory modules such as dual in-line memory modules (DIMMs), single in-line memory modules (SIMMs), small outline DIMMs (SO-DIMMs), and so on. The non-volatile storage 150 may include memory that retains its contents during a power loss, such as flash memory, memristor technology, and the like. The non-volatile storage 150 may include high speed random access memory, e.g., memory associated with faster performance than other forms of storage such as hard drives or solid state discs, and may include a portion of memory that is volatile.

[0012] The volatile memory modules 140 include persistent data 122 and non-persistent data 124. The persistent data 122 represents portion(s) of the volatile memory module 140 that are to be tracked and/or backed-up. The persistent data 122 may encompass a majority of a given volatile memory module 140, or may represent a small portion of the volatile memory module 140. Notably, examples are not limited to needing to perform a full or complete back-up of an entire given volatile memory module 140, in order to back-up the persistent data 122 existing in a volatile memory module 140. Examples described herein may use, e.g., tracking software (such as a metadata engine) to keep track of the areas of persistent data 122 that are to be backed-up. Such information regarding the persistent data 122 may be stored as metadata 120.

In response to a power loss condition (e.g., blackout or other interruption to power delivery to the computing system 100), the persistent data 122 may be moved/copied to the non-volatile storage 150. Thus, examples such as system 100 enable mapping of persistent data 122 and non-persistent data 124 within a single linear region of memory, which may be allocated to an application. Accordingly, the system 100 may use, e.g., commodity volatile DIMMs for a majority of the computing system's memory space, while providing a relatively smaller amount of non-volatile DIMMs to serve as non-volatile storage 150 for backed-up persistent data 126.

[0013] The persistent data 122 does not need to be from contiguous memory addresses, and may be located at disparate memory locations throughout the plurality of volatile memory modules 140. Thus, specific portions of the volatile memory modules 140 may be targeted (e.g., according to metadata 120) for back-up, whether located in block 1, or block 12, or other non-contiguous locations from various memory modules. In alternate examples, the persistent data 122 may occupy the entirety of a given volatile memory module 140. The volatile memory modules 140 are not limited to those located physically within a given computing system, and may be located throughout systems across multiple geographical locations. The backed-up persistent data 126 is shown as a single block of data, but may be spread as multiple blocks throughout multiple memory modules comprising the non-volatile storage 150.

[0014] The metadata 120 may be used by the system 100 to keep track of addresses of the persistent data 122 that is to be backed-up, and may serve as a data pointer. The metadata 120 is shown stored in storage 104, and in alternate examples, the metadata 120 may be stored, along with the pertinent persistent data, in the non-volatile storage 150. In some examples, the metadata 120 may be provided as a descriptor table, a linked list of descriptors, and so on.

[0015] For example, the system 100 may use a caching procedure to identify areas of the volatile memory modules 140 that are to be backed-up. In an example, such areas may be identified in the metadata 120 according to the memory engine 110 identifying data that has passed through the static random

access memory (SRAM) of the system's CPU. Other approaches may be used to flag persistent data 122 and store the corresponding identification information in the metadata 120. Such techniques may be applied dynamically in real-time during operation of the system 100. In alternate examples, the system 100 may periodically check for any updates to what data is to be considered persistent, and its corresponding locations etc.

[0016] Accordingly, use of metadata 120, to identify the persistent data 122 to be backed-up to non-volatile storage 150, can optimize the manner in which applications interact with system 100, by presenting memory to the applications as a whole memory space, e.g., in one continuous region, without a need to sector off the memory space into a volatile section(s) and a non-volatile section(s). Examples may thereby provide the memory space as a BIOS-to-application interface, enabling applications to indicate what data is to be backed up, and have the system 100 track that information in metadata 120 and save that pertinent information as backed-up persistent data 126 in non-volatile storage 150. Thus, a large variety of applications may seamlessly make use of non-volatile storage 150 in conjunction with the volatile memory modules 150, without a need to customize the application for dealing with segmented memory, because system 100 enables such interaction seamlessly.

[0017] System 100 may provide applications with a continuous block of memory that enjoys backed-up persistent data 126 on non-volatile storage 150. Memory map designs for using non-volatile storage/memory for persistent data can be simplified, enabling interaction and support for applications that would not otherwise support back-up of persistent data 122 onto non-volatile memory (e.g., due to a need for a single linear memory region that system 100 may provide, despite the presence of both volatile and non-volatile memory).

[0018] Thus, examples described herein enable multiple benefits and avoid underutilization of persistent storage/memory. Applications that otherwise could not, may now take advantage of persistent memory systems, and complex applications that have been modified to address segmented memory may be streamlined by avoiding a need to separately address volatile and non-volatile memory spaces. Furthermore, the non-volatile storage 150 may be used

without slowing down system and/or application performance, due to the high-speed nature of the non-volatile storage 150 (in contrast to a slower forms of non-volatile storage, such as a hard drive, for example).

[0019] Storage 104 may be accessible by the system 100, to serve as a computer-readable repository to store information such as metadata 120 that may be referenced by the memory engine 110 during operation of the system 100. As described herein, the term "controller" and/or "engine" may include electronic circuitry for implementing functionality consistent with disclosed examples. For example, memory engine 110 may represent combinations of hardware devices (e.g., processor and/or memory) and programming to implement the functionality consistent with disclosed implementations. In examples, the programming for the engines may be processor-executable instructions stored on a non-transitory machine-readable storage media, and the hardware for the engines may include a processing resource to execute those instructions. An example system (e.g., a computing device), such as system 100, may include and/or receive tangible non-transitory computer-readable media storing the set of computer-readable instructions. As used herein, the processor/processing resource may include one or a plurality of processors, such as in a parallel processing system, to execute the processor-executable instructions. The memory can include memory addressable by the processor for execution of computer-readable instructions. The computer-readable media can include volatile and/or non-volatile memory such as a random access memory ("RAM"), magnetic memory such as a hard disk, floppy disk, and/or tape memory, a solid state drive ("SSD"), flash memory, phase change memory, and so on.

[0020] In some examples, the functionality of engines may correspond to operations performed in response to, e.g., information from storage 104, interactions as interpreted and/or initiated by the, e.g., memory engine 110, and so on. The storage 104 may be accessible by the system 100 as a computer-readable storage media, in which to store items in a format that may be accessible by the memory engine 110. Thus, examples may be provided as a software mechanism.

[0021] FIG. 2 is a block diagram of a computing system 200 including memory controller instructions 210, DMA engine instructions 230, and metadata instructions 260 according to an example. The computer-readable media 204 is associated with a processor 202 and metadata 220, and may store the instructions 210, 230, 260. The memory controller instructions 210 may instruct a memory controller to perform data access/flow. The DMA engine instructions 230 may instruct a DMA engine to copy/move data between memory types. Accordingly, the memory controller instructions 210 and/or DMA engine instructions 230 may be carried out by the memory engine 110 of FIG. 1. The metadata instructions 260 may correspond to a metadata engine (not specifically shown in FIG. 1; see metadata engine 320 of FIG. 3) that may be included in the computing system 100 of FIG. 1. In some examples, operations performed when instructions 210, 230, and 260 are executed by processor 202 may correspond to the functionality of memory engine 110 and the metadata engine.

[0022] As set forth above with respect to FIG. 1, memory engine 110 may include combinations of hardware and programming that may correspond to memory controller instructions 210 and/or DMA engine instructions 230. Such components may be implemented in a number of fashions. For example, the programming may be processor-executable instructions stored on tangible, non-transitory computer-readable media 204 and the hardware may include processor 202 for executing instructions 210, 230, 260. Processor 202 may, for example, include one or multiple processors. Such multiple processors may be integrated in a single device or distributed across devices. Media 204 may store program instructions, that when executed by processor 202, implement system 100 of FIG. 1 (and/or system 300 of FIG. 3). Media 204 may be integrated in the same device as processor 202, or it may be separate and accessible to that device and processor 202.

[0023] In some examples, program instructions can be part of an installation package that when installed can be executed by processor 202 to implement system 100. In this case, media 204 may be a portable media such as a CD, DVD, flash drive, or a memory maintained by a server from which the

installation package can be downloaded and installed. In another example, the program instructions may be part of an application or applications already installed. Here, media 204 can include integrated memory such as a hard drive, solid state drive, or the like. While in FIG. 2, media 204 includes instructions 210, 230, 260, one or more instructions may be located remotely from media 204. Conversely, although FIG. 2 illustrates metadata 220 located separate from media 204, the metadata 220 may be included with media 204.

[0024] The computer–readable media 204 may provide volatile storage, e.g., random access memory for execution of instructions. The computer-readable media 204 also may provide non-volatile storage, e.g., hard disk or solid state disk for storage. Components of FIG. 2 may be stored in any type of computer-readable media, whether volatile or non-volatile. Content stored on media 204 may include images, text, executable files, scripts, or other content that may be used by examples as set forth below. For example, media 204 may contain instructions 210, 230, 260 and/or information that may be used by memory engine 110 (or other engines such as a metadata engine etc.), to provide control or other information.

[0025] FIG. 3 is a block diagram of a computing system 300 including a memory controller 310, a DMA engine 330, and a metadata engine 320 according to an example. The memory controller 310 may be associated with an allocation threshold 312. The DMA engine 330 and memory controller 310 may form at least a portion of a central processing unit (CPU) 302, which may receive a power loss signal 306. The DMA engine 330 and memory controller 310 may also correspond to a memory engine, such as memory engine 110 of FIG. 1. The DMA engine 330 and memory controller 310 are coupled to memory bus 314, which is coupled to a plurality of volatile DIMMs 340 and non-volatile DIMMs 350. Back-up energy source 303 is to provide power to the DMA engine 330, memory controller 310, volatile DIMMs 340, and non-volatile DIMMs 350 (the back-up energy source domain shown enclosed in dashed lines). The metadata engine 320 may interact with components of the CPU 302, DIMMs 340, 350 (via memory bus 314), BIOS 308, and operating system (OS) 309. The metadata engine 320 is associated with memory locations 328.

[0026] Examples based on system 300 may enable non-volatile memory back-up at a CPU/system level. The CPU 302 may be selectively powered, e.g., to power those portions of the CPU 302 used to move/copy at least a portion of the memory contents (e.g., persistent data) from the volatile DIMMs 340 to the non-volatile DIMMs 350, while allowing other portions of the CPU 302 to remain unpowered. Thus, there is no need for system 300 to use a customized application-specific integrated circuit (ASIC) or system on a chip (SOC) in order to control the memory operations, because system 300 is operable based on using a portion of the computing system's CPU 302 (e.g., the DMA engine 330 and the memory controller 310). System 300 thus may shut down unnecessary resources in the CPU 302, to reduce power during back-up operations. In an example, system 300 may start a back-up process by quiescing the CPU 302 and going into a lowest power state that still supports operation of the DMA engine 330 and the memory controller 310, to move persistent data to the non-volatile DIMMs 350. Thus, there is no need to repurpose signals/channels used to interact with the DIMMs, which may be accessed and controlled normally by the CPU 302 (or other memory controller 310/DMA Engine 330). Although shown integrated into the CPU 302, in alternate examples, the memory controller 310 and/or the DMA Engine 330 may be provided separately. In some examples, the back-up procedure may use compression and/or encryption on the persistent data to be backed up to non-volatile memory.

[0027] System 300 includes a plurality of memory slots that are occupied by the volatile and non-volatile DIMMs 340, 350. As shown, a relatively small portion of the DIMM slots are occupied by non-volatile DIMMs 350. However, system 300 enables non-volatile memory back-up support across the memory address map, even though the memory slots are populated mostly by volatile DIMMs 340 that do not have flash memory (e.g., standard or commodity DIMMs) in the example of FIG. 3. The persistent data may be distributed throughout the DIMMs 340, 350, e.g., according to the memory locations 328 of metadata. After a back-up procedure is performed, the persistent data may reside safely backed-up in the non-volatile DIMMs 350. Although illustrated as

two non-volatile DIMMs 350 located in the third and tenth memory slots, examples may use fewer or greater numbers of non-volatile DIMMs, which may be located in other memory slot(s).

[0028] Thus, examples do no need to locate non-volatile memory at every or at specific regions of the memory map, thereby reducing system costs by avoiding a need to populate all memory slots with non-volatile DIMMs 350. For example, an application to be executed on system 300 may be allocated the use of two volatile DIMMs 340 for working memory. If only a small portion of the data in either of those two volatile DIMMs 340 is persistent data (as indicated by metadata/memory locations 328), there is no need to back-up the entirety of those two volatile DIMMs 340. Rather, system 300 may back-up just the persistent data located on the allocated two volatile DIMMs 340, which may fit on a fraction of a single non-volatile DIMM 350.

[0029] Example systems are not limited to removable memory modules, and may include fixed/soldered memory or other memory that is not in the form of a memory module. Example systems 300 may include memory slots that support volatile DIMMs 340, non-volatile DIMMs 350, and/or both volatile and non-volatile DIMMs 340 and 350. In an example, a system's BIOS may discover what types of memory slots and/or memory modules are populating the system's memory slots, and/or may refer to a configuration process performed by the customer to designate a given memory configuration (e.g., what type and how much of the memory is available as non-volatile storage). Such information may be used by the BIOS, and/or communicated to an operating system (OS), to perform allocations according to available memory. In an example system, every n^{th} memory slot may provide support for non-volatile memory modules, enabling the system to support a fewer number of non-volatile memory modules than volatile memory modules, thereby saving costs on the system motherboard (compared to enabling support at all of the memory slots). Such configurations may be supported according to desired memory mapping and corresponding capabilities of the memory controller 310 and/or DMA engine 330. In an example, the non-volatile memory may be provided at a top of a linear memory stack, e.g., to provide good performance in backing up the data efficiently and

quickly by avoiding segmentation of the memory, in view of a limited supply of back-up energy source 303 to perform the back-up. Arranging the non-volatile DIMMs 350 at the top of memory further supports techniques for improving memory performance such as interleaving etc., and enables the OS to exclude the non-volatile DIMMs 350 for such operational performance enhancements to the volatile DIMMs 340. In examples, a fixed number of memory slots may be designated/indicated as supporting non-volatile DIMMs 350, and in other examples the memory slots may dynamically support any arrangements of volatile and non-volatile DIMMs 340, 350.

[0030] The back-up energy source 303 may be used to perform back-ups, e.g., in response to a power loss condition. For example, the CPU 302 may receive a power loss signal 306 indicating a power loss. Upon loss of power to the system 300, the back-up energy source 303 may serve as a local finite power source to provide enough energy to continue to allow the CPU's memory controller 310 and DMA engine 330 to transfer the persistent data from the volatile DIMMs 340 to the non-volatile DIMM(s) 350. Thus, the back-up energy source 303 may provide enough energy to power the memory controller 310, the DMA engine 330, the volatile DIMMs 340 (e.g., those associated with the persistent data, which may be a subset of the volatile DIMMs 340), and the non-volatile DIMMs 350 (e.g., those associated with a capacity to store the persistent data to be backed up, which may be a subset of the available non-volatile DIMMs 350). Additional modules (not specifically shown) within the CPU 302 may also be powered as needed, or may share the same power delivery system as the memory controller 310 and/or DMA engine 330. The back-up energy source 303 may be chosen to provide power for a finite period of time, e.g., that amount of time needed to transfer persistent data from volatile DIMMs 340 to a flash component on a non-volatile DIMM 350.

[0031] In an example, the back-up energy source 303 may be integrated with a power delivery circuitry on a system motherboard, and may be provided as a battery, capacitor, and so on. In alternate examples, the back-up energy source 303 may be integrated with a system power supply (not shown). A capacity of the back-up energy source 303 may be chosen based on

parameters/considerations including: 1) the total power needs of the CPU 302 and DIMMs 340,350 during the data transfer process; 2) the finite period of time which back-up power is to be available; 3) the maximum number of non-volatile DIMMs 350 to be supported; 4) any additional signals needed on the DIMM connector interface and/or which memory pins are repurposed, if applicable; and 5) whether a subset of the DIMM slots will support the non-volatile DIMMs 350 (static assignment) or if all slots will support the feature (dynamic). Even though a CPU 302 may have a high energy rating, the given percentage of persistent data to non-persistent data may be relatively small (e.g., as reflected by the relatively low number of non-volatile DIMMs 350). Thus, even though the CPU 302 may need a relatively large amount of energy typically, such needs would only exist for a relatively short time. Also, a small portion of the total CPU 302 may actually be needed to perform the transfer, so the actual CPU power draw is substantially reduced during the example data movement/memory transfers.

[0032] The metadata engine 320 may be provided as an executable application to keep track of what information is to be treated as persistent data and backed up from the volatile DIMMs 340 to the non-volatile DIMMs 350. The metadata engine 320 may communicate such information to the memory controller 310 (or other software to be performing the back-up), to back up the information identified in the memory locations 328 of the metadata. In an example, the metadata engine 320 may push addresses of the memory locations 328 to be backed up, and even the metadata itself, into the non-volatile DIMM(s) 350.

[0033] The metadata engine 320 may communicate with, be part of, and/or be operated by the BIOS 308 and/or the operating system 309 associated with the system 300. The metadata engine 320 may make calls to the BIOS 308 and/or OS 309. In alternate examples, the metadata engine 320 may interact with the CPU 302 operating in a system management mode (SMM) to run in the background as an out-of-band operation outside of the OS context. Thus, the metadata engine 320 may identify the memory locations 328 corresponding to data to be backed up. Accordingly, the metadata engine 320 may collect

information dynamically during system operations, such that the collected metadata information is ready for use in response to a power loss condition as indicated by the power loss signal 306.

[0034] The memory controller 310 may cause the system 300 to operate within the constraints of the back-up energy source 303, by applying an allocation threshold 312 to operation of the system 300. For example, the OS 309 and/or BIOS 308 may employ a "throttle bit" under direction of the memory controller 310 according to the allocation threshold 312. The memory controller 310 may allocate out a certain amount of memory to applications, limited according to available energy capacity of the back-up energy source 303 and power needs of a system to reliably back-up the certain amount memory that has been allocated. In an example, the system 300 may impose the allocation threshold 312 across the memory map to apply to memory usage altogether. Once the allocation threshold 312 is reached, the memory controller 310 may signal that it will no longer allocate more persistent memory, due to a lack of expected power capacity at the back-up energy source 303 to reliably back-up further allocated memory. Such information may be strictly enforced, or may be used to generate a notification to the user that the allocation threshold 312 has been reached and further allocation may be at risk. A given system 300 may identify how much time would be needed to back-up a given amount of data, how much back-up energy source 303 capacity is available, and how much time will be needed to back-up the given amount of data, to establish an allocation threshold 312. In examples, the BIOS 308 and/or OS 309 may manage the allocation threshold 312 and its enforcement, as each application is requesting more persistent memory. Such enforcement may be performed in real-time (i.e., during run time before a power loss condition occurs), to avoid allocation of too much data that would not fit within the window of opportunity provided by the capacity of a given back-up energy source 303.

[0035] Referring to Figure 4, a flow diagram is illustrated in accordance with various examples of the present disclosure. The flow diagram represents processes that may be utilized in conjunction with various systems and devices as discussed with reference to the preceding figures. While illustrated in a

particular order, the disclosure is not intended to be so limited. Rather, it is expressly contemplated that various processes may occur in different orders and/or simultaneously with other processes than those illustrated.

[0036] FIG. 4 is a flow chart 400 based on copying persistent data according to an example. In block 410, a memory controller engine is to identify, according to metadata, persistent data to be backed up from a plurality of volatile memory modules including non-persistent data that is not to be backed up. For example, in a back-up operation, the memory controller engine may disregard a portion of the data in the volatile memory modules, and back-up the persistent data according to the address locations indicated in the metadata. In block 420, a direct memory access (DMA) engine is to copy the persistent data from the plurality of volatile memory modules to the non-volatile storage, excluding the non-persistent data. For example, by excluding the non-persistent data, the DMA engine may copy a portion of data from a relatively large number of volatile memory modules that happen to contain persistent data, and move that data to a relatively small number of non-volatile memory module(s) that form the non-volatile storage.

[0037] Examples provided herein may be implemented in hardware, programming, or a combination of both. Example systems can include a processor and memory resources for executing instructions stored in a tangible non-transitory computer-readable media (e.g., volatile memory, non-volatile memory, and/or computer-readable media). Non-transitory computer-readable media can be tangible and have computer-readable instructions stored thereon that are executable by a processor to implement examples according to the present disclosure. The term “engine” as used herein may include electronic circuitry for implementing functionality consistent with disclosed examples. For example, memory engine 110 of FIG. 1 may represent combinations of hardware devices and programming to implement the functionality consistent with disclosed implementations. In some examples, the functionality of engines may correspond to operations performed by user actions, such as selecting steps to be executed by processor 202 (described above with respect to FIG. 2).

WHAT IS CLAIMED IS:

1. A computing system comprising:

a memory engine to identify, according to metadata, persistent data to be backed up from a plurality of volatile memory modules, wherein a given volatile memory module of the plurality of volatile memory modules includes persistent data to be backed up and non-persistent data that is not to be backed up;

wherein the memory engine is to copy the persistent data from the plurality of volatile memory modules to the non-volatile storage, and exclude the non-persistent data.

2. The computing system of claim 1, further comprising a back-up energy source to power the memory engine, the plurality of volatile memory modules, and the non-volatile storage in response to a power loss condition, to enable the memory engine to copy the persistent data during a power loss condition.

3. The computing system of claim 2, wherein the memory engine is to identify an allocation threshold for persistent data memory allocation, corresponding to a capability of the back-up energy source to successfully back-up an amount of the persistent data allocated under the allocation threshold.

4. The computing system of claim 1, wherein the plurality of volatile memory modules and the non-volatile storage are comprised of dual in-line memory modules (DIMMs), and wherein the persistent data is copied to a number of the non-volatile storage DIMM(s) fewer in number than the volatile memory DIMMs from which the persistent data is copied.

5. The computing system of claim 4, further comprising a plurality of DIMM slots to receive the DIMMs, wherein a portion of the DIMM slots are compatible with the non-volatile storage DIMMs, and a portion of the DIMM slots are compatible with the volatile memory DIMMs.

6. The computing system of claim 1, wherein the persistent data identified by the memory engine coexists in a given volatile memory module with the non-persistent data in a single contiguous block, such that a memory space address map of the computing system is presented as a contiguous block including persistent and non-persistent components.

7. The computing system of claim 1, wherein the memory engine is to copy the persistent data from non-contiguous portions of the plurality of volatile memory modules, and write the persistent data to the non-volatile storage in a contiguous sequential manner.

8. The computing system of claim 1, further comprising a metadata engine to track memory locations of the persistent data in response to the persistent data being written to the plurality of volatile memory modules, and store the memory locations in the metadata for locating the persistent data.

9. The computing system of claim 1, where the metadata is collected by at least one of the computing system's Basic Input/Output System (BIOS), the computing system's Operating System (OS), and the computing system's central processing unit (CPU) operated in system management mode (SMM).

10. The computing system of claim 9, wherein the CPU, in response to a power loss signal, is to quiesce by entering a low power state, and copy the persistent data from the volatile memory modules to the non-volatile storage.

11. The computing system of claim 1, wherein the metadata is stored in a portion of the non-volatile storage.

12. A method, comprising:

identifying, by a memory engine according to metadata, persistent data to be backed up from a plurality of volatile memory modules, wherein a given

volatile memory module of the plurality of volatile memory modules includes persistent data to be backed up and non-persistent data that is not to be backed up; and

copying, by the memory engine, the persistent data from the plurality of volatile memory modules to the non-volatile storage, excluding the non-persistent data.

13. The method of claim 12, further comprising tracking, by a metadata engine, memory locations of the persistent data in response to the persistent data being written to the plurality of volatile memory modules; and

storing the memory locations in the metadata for locating the persistent data.

14. A non-transitory machine-readable storage medium encoded with instructions executable by a computing system that, when executed, cause the computing system to:

identify, according to metadata, persistent data to be backed up from a plurality of volatile memory modules, wherein a given volatile memory module of the plurality of volatile memory modules includes persistent data to be backed up and non-persistent data that is not to be backed up; and

copy, by a memory engine, the persistent data from the plurality of volatile memory modules to the non-volatile storage, excluding the non-persistent data.

15. The storage medium of claim 14, further comprising instructions that cause the computing system to identify a power loss condition, and power, by a back-up energy source, the memory engine, the plurality of volatile memory modules, and the non-volatile storage, to enable the memory engine to copy the persistent data.

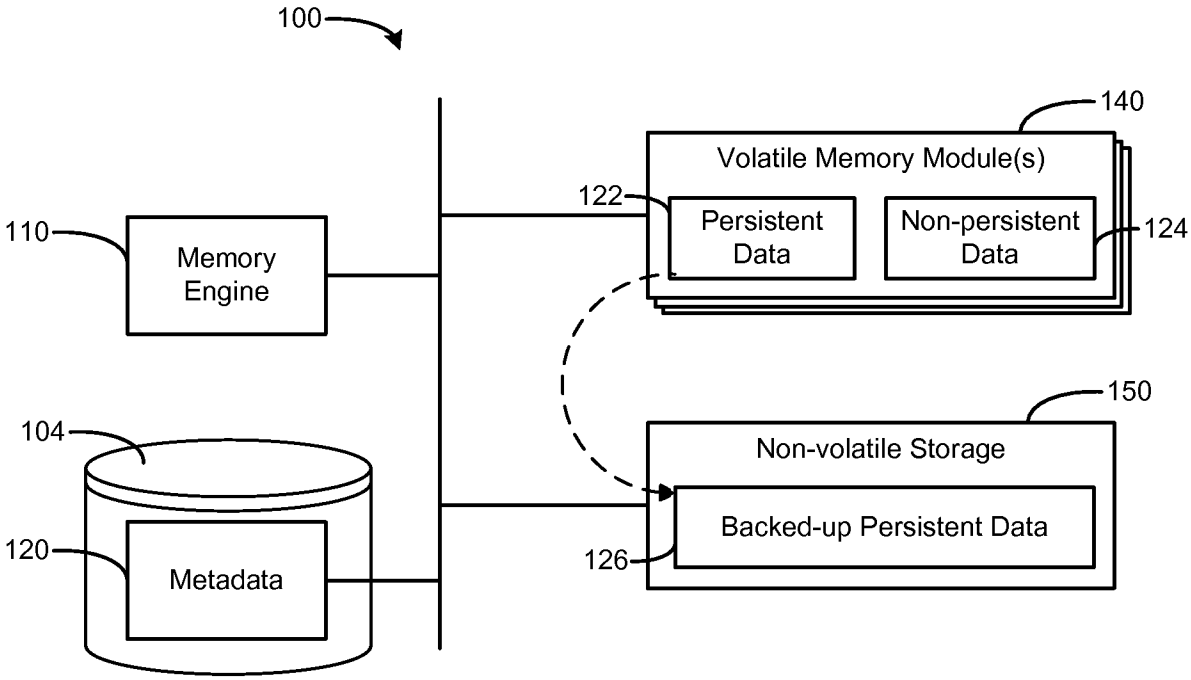
100
140
Volatile Memory Module(s)
122
Persistent Data
Non-persistent Data
124
110
Memory Engine
150
104
Non-volatile Storage
Backed-up Persistent Data
120
Metadata
126

FIG. 1

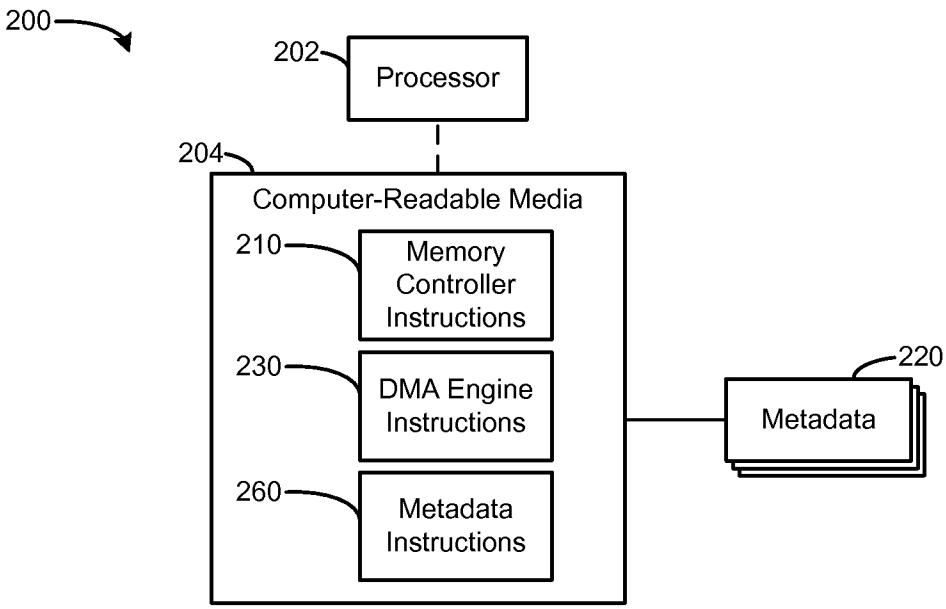
200
202
Processor
204
Computer-Readable Media
210
Memory Controller Instructions
230
DMA Engine Instructions
260
Metadata Instructions
220
Metadata

FIG. 2

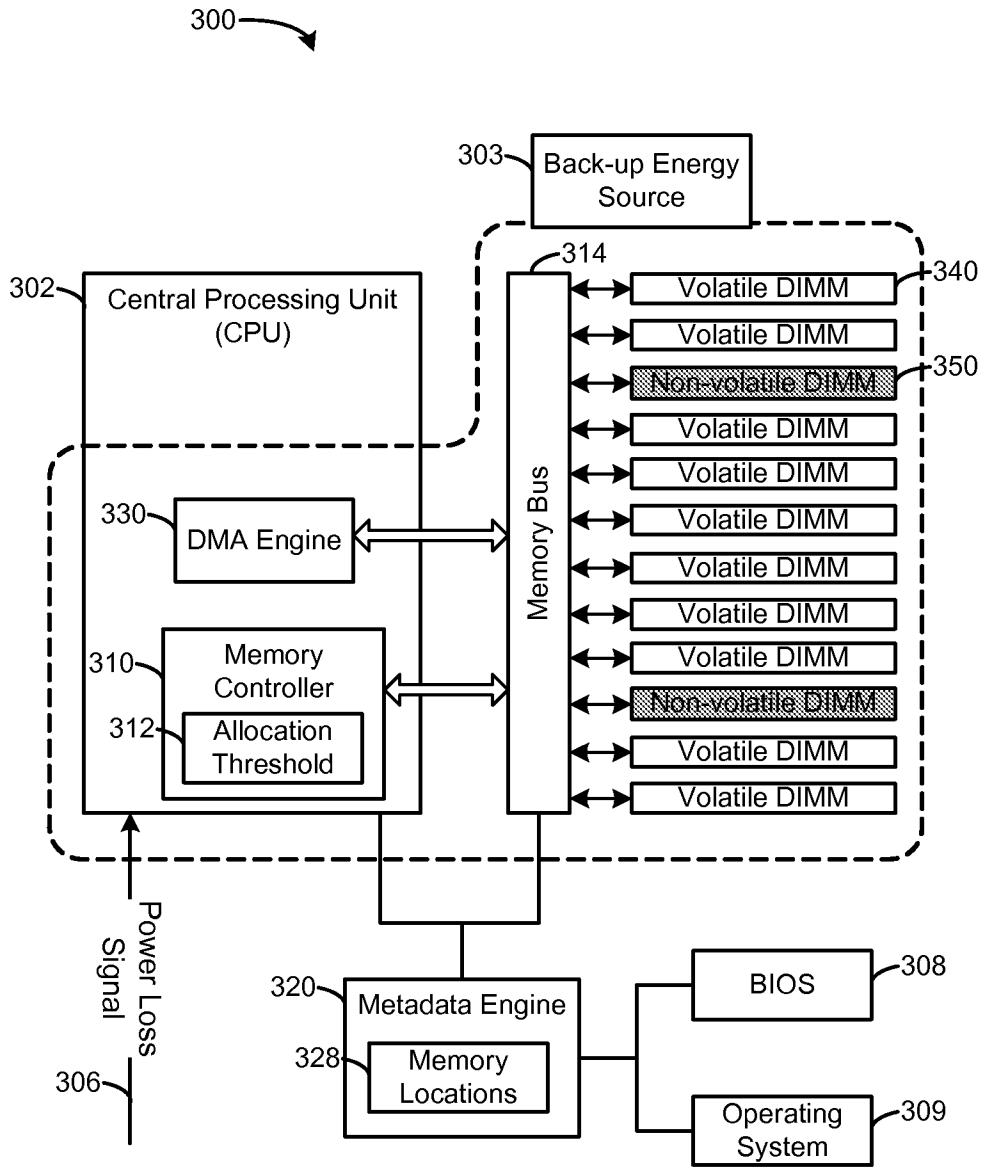
300
303
Back-up Energy Source
314
302
Central Processing Unit (CPU)
340
Volatile DIMM
Volatile DIMM
350
Non-volatile DIMM
Volatile DIMM
Volatile DIMM
Volatile DIMM
Volatile DIMM
Volatile DIMM
Volatile DIMM
Non-volatile DIMM
Volatile DIMM
Volatile DIMM
Memory Bus
330
DMA Engine
310
Memory Controller
312
Allocation Threshold
Power Loss Signal
306
320
Metadata Engine
328
Memory Locations
BIOS
308
Operating System
309

FIG. 3

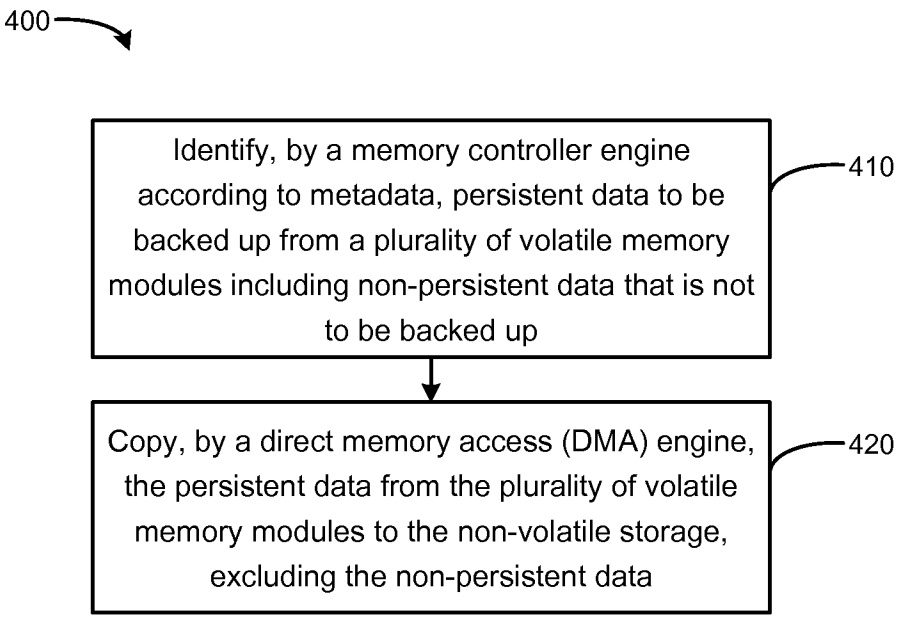
400
Identify, by a memory controller engine according to metadata, persistent data to be backed up from a plurality of volatile memory modules including non-persistent data that is not to be backed up
410
Copy, by a direct memory access (DMA) engine, the persistent data from the plurality of volatile memory modules to the non-volatile storage, excluding the non-persistent data
420

FIG. 4

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2015/013213

A. CLASSIFICATION OF SUBJECT MATTER

G06F 12/16(2006.01)i, G06F 11/14(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F 12/16; G06F 17/30; G06F 13/38; G06F 13/14; G06F 13/20; G06F 1206; G06F 11/22; G06F 13/00; G06F 11/14

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Korean utility models and applications for utility models
Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
eKOMPASS(KIPO internal) & Keywords: persistent, data, backup, volatile, nonvolatile, copy

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2014-0195564 A1 (FUSION-IO, INC.) 10 July 2014 See paragraphs [0006], [0011], [0036], [0056], [0060], [0071]-[0072], [0090], [0111], [0161], [0167], [0281]; claim 11; and figures 2-4.	1-3,6-15
Y		4-5
Y	US 2012-0059967 A1 (JONATHAN R. HINKLE et al.) 08 March 2012 See paragraphs [0007], [0043], [0050]; and figures 1A, 2.	4-5
A	WO 2014-003764 A1 (HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P.) 03 January 2014 See paragraphs [0010], [0017]; and figure 1.	1-15
A	US 6842823 B1 (THOMAS M. OLSON) 11 January 2005 See column 4, lines 33-44; column 6 lines 4-18; claim 22; and figure 2.	1-15
A	US 2014-0215277 A1 (INTERNATIONAL BUSINESS MACHINES CORPORATION) 31 July 2014 See paragraphs [0014], [0019]; and figure 1.	1-15

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent 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 when the document is taken alone
"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

Date of the actual completion of the international search	Date of mailing of the international search report
17 September 2015 (17.09.2015)	**07 October 2015 (07.10.2015)**

Name and mailing address of the ISA/KR
International Application Division
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan City, 302-701,
Republic of Korea
Facsimile No. +82-42-472-7140

Authorized officer
BYUN, Sung Cheal
Telephone No. +82-42-481-8262

Form PCT/ISA/210 (second sheet) (January 2015)

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.
PCT/US2015/013213

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2014-0195564 A1	10/07/2014	None	
US 2012-0059967 A1	08/03/2012	US 08583869 B2	12/11/2013
		US 08656072 B2	18/02/2014
		US 2011-153903 A1	23/06/2011
		US 2012-059970 A1	08/03/2012
		US 2012-159045 A1	21/06/2012
		WO 2011-087820 A2	21/07/2011
		WO 2011-087820 A3	10/11/2011
		WO 2011-087820 A8	22/03/2012
WO 2014-003764 A1	03/01/2014	CN 104246732 A	24/12/2014
		EP 2867779 A1	06/05/2015
		KR 10-2015-0032659 A	27/03/2015
		US 2015-0127890 A1	07/05/2015
US 6842823 B1	11/01/2005	AU 2001-51617 A1	30/10/2001
		AU 2002-307264 A1	03/12/2002
		AU 2002-307264 A8	03/12/2002
		EP 1277115 A2	22/01/2003
		US 06802022 B1	05/10/2004
		US 06862689 B2	01/03/2005
		US 06901481 B2	31/05/2005
		US 2001-0032300 A1	18/10/2001
		US 2002-0016935 A1	07/02/2002
		US 2002-0152429 A1	17/10/2002
		WO 2001-080008 A2	25/10/2001
		WO 2001-080008 A3	06/06/2002
		WO 2002-095628 A2	28/11/2002
		WO 2002-095628 A3	12/02/2004
US 2014-0215277 A1	31/07/2014	GB 2510180 A	30/07/2014