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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
14 November 2019 (14.11.2019)

WIPO | PCT

(10) International Publication Number
WO 2019/217066 A1

(51) International Patent Classification:
G06F 12/0862 (2016.01) *G06F 12/02* (2006.01)

(21) International Application Number:
PCT/US2019/028688

(22) International Filing Date:
23 April 2019 (23.04.2019)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
15/975,614 09 May 2018 (09.05.2018) US
16/105,545 20 August 2018 (20.08.2018) US

(71) Applicant: **MICRON TECHNOLOGY, INC.** [US/US]; 8000 S. Federal Way, Boise, Idaho 83716-9632 (US).

(72) Inventors: **HASBUN, Robert Nasry**; 11425 Chula Vista Ct., San Jose, California 95127 (US). **GANS, Dean, D.**; 8000 S. Federal Way, Boise, Idaho 83716-9632 (US). **DARUWALLA, Sharookh**; 8000 S. Federal Way, Boise, Idaho 83716-9632 (US).

(74) Agent: **HARRIS, Philip**; Holland & Hart LLP, P.O. Box 11583, Salt Lake City, Utah 84147 (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, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, 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.

(54) Title: PREFETCH MANAGEMENT FOR MEMORY

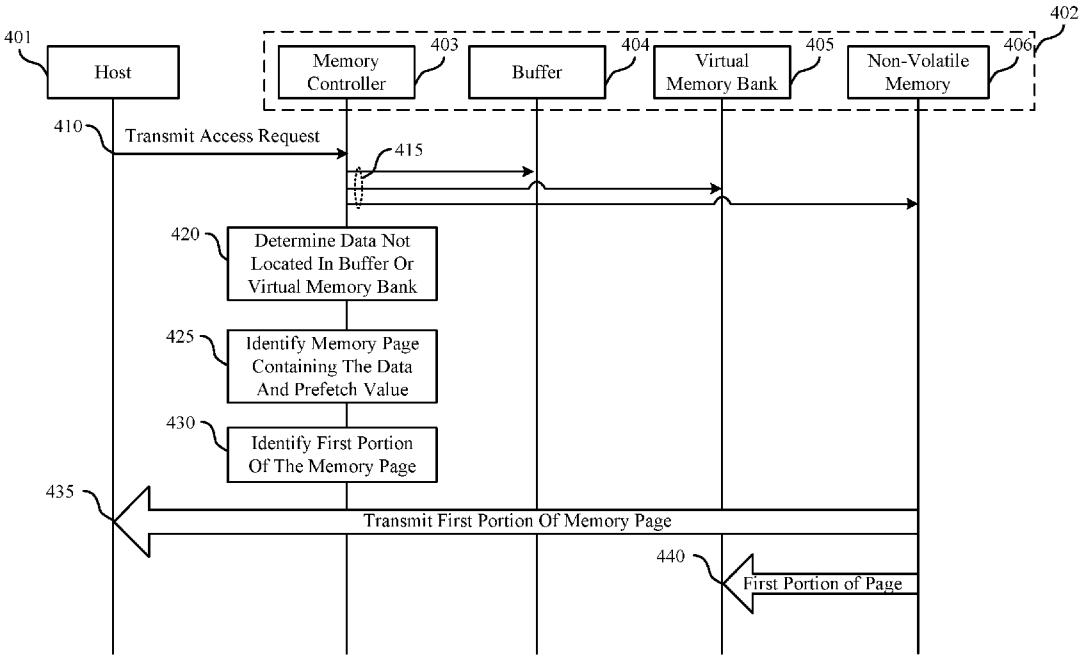

FIG. 4A

400-a

(57) Abstract: Methods, systems, and devices are described for wireless communications. A request for data located in a memory page of a memory array may be received at a device, and a value of a prefetch counter associated with the memory page may be identified. A portion of the memory page that includes the requested data may then be communicated between a memory array and memory bank of the device based on the value of the prefetch counter. For instance, the portion of the memory page may be selected based on the value of the prefetch counter. A second portion of the memory page may be communicated to a buffer of the device, and the value of the prefetch counter may be modified based on a relationship between the first portion of the memory page and the second portion of the memory page.

WO 2019/217066 A1

[Continued on next page]

(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).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

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

PREFETCH MANAGEMENT FOR MEMORY

CROSS REFERENCE

[0001] The present Application for Patent claims priority to U.S. Patent Application No. 15/975,614 by Hasbun et al., entitled "Prefetch Management For Memory," filed May 9, 2018, and U.S. Patent Application No. 16/105,545 by Hasbun et al., entitled "Prefetch Management For Memory," filed August 20, 2018, each of which is assigned to the assignee hereof and is expressly incorporated by reference in its entirety herein.

BACKGROUND

[0002] The following relates generally to memory systems and more specifically to prefetch management for memory.

[0003] A memory system may include various kinds of memory devices and controllers, which may be coupled via one or more buses to manage information in numerous electronic devices such as computers, wireless communication devices, internet of things devices, cameras, digital displays, and the like. Memory devices are widely used to store information in such electronic devices. Information may be stored in a memory device by programing different states of one or more memory cells within the memory device. For example, a binary memory cell may store one of two states, often denoted as a logic "1" or a logic "0." Some memory cells may be able to store more than two states.

[0004] Various types of memory devices exist, including magnetic hard disks, random access memory (RAM), read only memory (ROM), dynamic RAM (DRAM), synchronous dynamic RAM (SDRAM), ferroelectric RAM (FeRAM), magnetic RAM (MRAM), resistive RAM (RRAM), flash memory, phase change memory (PCM), and others. Memory devices may be volatile or non-volatile. Non-volatile memory cells may maintain their stored logic state for extended periods of time even in the absence of an external power source. Volatile memory cells, e.g., DRAM cells, may lose their stored logic state over time unless they are periodically refreshed by an external power source.

[0005] Improving memory systems, generally, may include reducing system power consumption, increasing memory system capacity, improving read/write speeds, providing non-volatility by use of persistent main memory, or reducing manufacturing costs at a certain performance point, among other metrics. Memory page size may affect system power and

power used to access a memory device. Moreover, system power may be affected or increased if memory accesses are not tailored for particular systems.

BRIEF DESCRIPTION OF THE DRAWINGS

[0006] FIG. 1 shows a diagram of a system including a non-volatile memory system or sub-system that supports prefetch management for main memory in accordance with examples of the present disclosure.

[0007] FIG. 2 illustrates an exemplary memory system or sub-system that supports prefetch management for main memory in accordance with examples of the present disclosure.

[0008] FIG. 3 illustrates an exemplary data structure and state diagram that support prefetch management for main memory in accordance with examples of the present disclosure.

[0009] FIGs. 4A and 4B illustrate exemplary process flows for a prefetch operation in accordance with examples of the present disclosure.

[0010] FIG. 5 shows a block diagram of a device that supports prefetch management for main memory in accordance with examples of the present disclosure.

[0011] FIGs. 6 through 8 show flowcharts illustrating a method for prefetch management for main memory in accordance with examples of the present disclosure.

DETAILED DESCRIPTION

[0012] A memory system may utilize different memory technologies to store data. For example, a memory system may include a first memory array (which may be a volatile memory array, like DRAM) and a second memory array (which may be a non-volatile memory array, like FeRAM or 3D XPoint™). The volatile memory array may perform communications with an external processor using less latency than the non-volatile memory array. Also, in some cases, access operations (e.g., read/write operations) performed on the volatile memory array may consume less power than access operations performed on the non-volatile memory array. The non-volatile memory array, however, may reduce long-term power consumption relative to the volatile memory array—e.g., by reducing or eliminating various refresh operations. And unlike the volatile memory array, the non-volatile memory array may retain stored data if the non-volatile memory array loses power. In some cases, the

memory system may be configured and operable to leverage beneficial characteristics of both of these technologies.

[0013] For example, the memory system may store a majority of data in the non-volatile memory array and may strategically move data to the volatile memory array. In some cases, the memory system may preemptively move data stored in the non-volatile memory array and likely to be accessed to the volatile memory array before receiving an access request for the data. This operation may be referred to as a "prefetch" operation. During a prefetch operation, the memory system may move, from the non-volatile memory array to the volatile memory array, one or more pages of data (or a "page" or "memory page") that are likely to be accessed. In this way, the memory system may reduce access operations performed on the non-volatile memory array, and thus, increase data throughput of the memory system by leveraging the relatively faster timing associated with accessing the volatile memory array.

[0014] Additionally or alternatively, the memory system may perform partial prefetch operations. For example, the memory system may prefetch a portion of a memory page as opposed to a full memory page. In some cases, the memory system may prefetch a portion of a memory page based on prior access operations performed on data stored in the non-volatile memory array (i.e., based on access history). In this way, the non-volatile memory array may determine which portions of the memory page are likely to be accessed based on previous access operations, and may provide the portions of a memory page that are likely to be accessed to the volatile memory array. For example, the memory system may determine that two portions of a non-volatile memory page are often accessed together and may cause the non-volatile memory array to deliver, to the volatile memory array, data corresponding to both portions of the non-volatile memory page when either portion of the memory page is accessed. In some cases, a memory page may be partitioned into memory subpages (or "subpages") and a portion of a memory page may correspond to one or more subpages.

[0015] As discussed above, a memory system may include a volatile memory array and a non-volatile memory array. The volatile memory array may be configured with one or more memory pages having a first size, and a device may store data in the memory array according to the page size. In some examples, the volatile memory array may be a dynamic random access memory (DRAM) array and may use memory pages having a page size of 2048 bytes, for example. In some cases, a device may request data stored in the volatile memory array to support the operation of a user application related to the device. For example, a processor at

the device may initiate a read operation at the volatile memory array, and the volatile memory array may provide the requested data to the processor. The volatile memory array may identify a memory page that includes the requested data and may access the entire memory page before providing the requested data to the processor. The processor may communicate with the volatile memory array according to a certain protocol (e.g., DRAM memory access protocols) that is associated with specified voltage and timing requirements.

[0016] The non-volatile memory array may store data in memory pages using a second predetermined size or a variable size. The non-volatile memory array may in some cases use smaller page sizes (e.g., 64 bytes, 128 bytes, 192 bytes, or 256 bytes) than the volatile array. Moreover, unlike a DRAM array, the non-volatile memory array may access one or more portions (e.g., multiple 64 byte portions) of a memory page without disturbing the logic states of other memory cells in the portion. Accessing fewer memory cells during a read operation may decrease power consumption of the non-volatile memory array. The non-volatile memory array may, however, be incompatible with certain memory access protocols (e.g., DRAM memory access protocols). For instance, the non-volatile memory array may be accessed using different voltages and/or timing than a DRAM array. Thus, the non-volatile memory array may not be directly accessible by a processor that is configured to communicate using DRAM memory access protocols.

[0017] The non-volatile memory array, despite having a smaller page size than the volatile memory array, may have a larger storage capacity than the volatile memory array. A processor that operates according to certain memory access protocols may have direct access to the volatile memory array and indirect access to the non-volatile memory array (e.g., through a memory controller). In some examples, the processor may be configured to communicate with the memory system as a single memory array and sends access requests using a first memory access protocol (e.g., a DRAM memory access protocol). A memory controller may interpret access requests from the processor and provide the requested data to the processor. In some cases, data stored in the volatile memory array may be accessed using and within the specifications of the memory access protocol of the processor, while data stored in the non-volatile memory array may not be accessible within the specification of the memory access protocol used by the processor—e.g., due to the non-volatile memory array using an alternative memory technology.

[0018] An indication that there will be a delay in acquiring the requested data may be sent to the processor when the memory system is unable to retrieve the requested data within a designated period of time. For example, the memory controller may indicate a read delay when the processor requests data that is stored in the non-volatile memory array and not in the volatile memory array—e.g., due to slower timing for accessing data in the non-volatile memory array. The processor may wait an additional amount of time for the data after receiving the indication, and the requested data may be provided to the processor according to the modified timing. In some examples, the requested data may also be moved to the volatile memory array in anticipation of receiving a subsequent access request for the data from the processor.

[0019] As discussed above and in some examples, certain memory pages stored in the non-volatile memory array may be preemptively moved to the volatile memory array to avoid future access delays before the data otherwise located in the memory pages is requested (i.e., the data may be prefetched). In some examples, access patterns for data stored in the non-volatile memory array are monitored to dynamically determine which memory pages to move to the volatile memory array—e.g., to determine which memory pages contain data that is likely to be accessed within a certain period of time. For example, a counter (which may be referred to as a "saturation counter") may be used to monitor the number of times data in a certain memory page is accessed within a certain period of time or cycle, and the memory page may be moved from the non-volatile memory array to the volatile memory array based on a value of the counter.

[0020] Power consumption may be further reduced by prefetching portions of a memory page instead of prefetching an entire memory page. A prefetched portion of the memory page may be selected based on the access history of data in the memory page. As discussed above, the prefetched portion may include one or more subpages of the memory page. For instance, portions of a memory page that are frequently accessed or that have been accessed a certain number of times may be prefetched. In some examples, a value of a counter (which may be referred to as a "prefetch counter") may be used to indicate how many and/or which subpages of a memory page to move from the non-volatile memory array to the volatile memory array. Each memory page may be associated with a prefetch counter, and the value of a respective prefetch counter may be adapted based on access history, or access patterns, or both, of the respective memory page. In some examples, the memory controller may include a respective

saturation counter for each of the subpages and the value of the prefetch counter may be based on the values of the saturation counters.

[0021] In some examples, a portion of a memory page may be communicated from a non-volatile memory array to a volatile memory array based on receiving a request for data located in the memory page and a value of a prefetch counter. For example, a request for data from a processor may be received and a memory page of the non-volatile memory array that contains the data may be identified. In some examples, the subpage that includes the requested data and a value of a prefetch counter associated with the memory page may also be identified. The identified subpage and the value of the prefetch counter may be used to determine which portion of the memory page, or a size of a portion of the memory page, to communicate between the non-volatile memory array and the volatile memory array.

[0022] In some examples, a memory system may include a memory buffer (or "buffer"), in addition to the volatile memory array and the non-volatile memory array. In some cases, the buffer may also be a volatile memory array. The memory system may update a value of a prefetch counter based on which subpages are transmitted to the buffer. For instance, a portion of a memory page including three subpages of data (which may correspond to 192 bytes of data) may be transmitted to the volatile memory array based on a value of a corresponding prefetch counter. Subsequently, a number of access operations performed on each of the subpages at the volatile memory array may be monitored. In some examples, data corresponding to only two of the three subpages may be transmitted from the volatile memory array to the buffer and the value of the corresponding prefetch counter may be modified. For example, the prefetch counter may be modified so that a portion of the memory page including two subpages of data (which may correspond to 128 bytes of data), instead of three subpages, is transmitted from the non-volatile memory array to the volatile memory array the next time the memory page is accessed. In some cases, the prefetch counter value is not updated until the subpages stored in the buffer are evicted.

[0023] Features of the disclosure introduced above are further described below at an exemplary system level in the context of FIG. 1. Specific examples of memory systems and operations are then described in the context of FIGs. 2 through 4. These and other features of the disclosure are further illustrated by and described with reference to the apparatus diagram of FIG. 5, which describe various components related controllers, as well as flowcharts of FIGs. 6 through 8, which relate to operations of prefetch management for main memory.

[0024] **FIG. 1** shows a diagram of a system 100 including a memory system or sub-system that supports prefetch management for main memory in accordance with examples of the present disclosure. System 100 may include device 105. Device 105 may include an interface controller 120, an SoC or processor 130, and various memory devices 170, 175, and 180. Device 105 may also include an input/output controller 135, a basic input/output system (BIOS) component 140, a board support package (BSP) 145, peripheral component(s) 150, and a direct memory access controller (DMAC) 155. The components of device 105 may be in electronic communication with one another through a bus 110.

[0025] Device 105 may be a computing device, electronic device, mobile computing device, or wireless device. Device 105 may be a portable electronic device. For example, device 105 may be a computer, a laptop computer, a tablet computer, a smartphone, a cellular phone, a wearable device, an internet-connected device, or the like. In some examples, device 105 may be configured for bi-directional wireless communication via a base station or access point. In some examples, device 105 may be capable of machine-type communication (MTC), machine-to-machine (M2M) communication, or device-to-device (D2D) communication. Device 105 may be referred to as a user equipment (UE), station (STA), mobile terminal, or the like.

[0026] Interface controller 120 may be configured to interface with SoC/processor 130. Interface controller 120 may also be configured to interface with various memory devices 170, 175, 180, or any combination thereof.

[0027] SoC/processor 130 may be configured to operate with various memory devices 170, 175, 180, or any combination thereof—either directly or via interface controller 120. SoC/processor 130 may also be referred to as a host, and may include a host controller. In some cases, SoC/processor 130 may perform some or all of the functions of interface controller 120 described herein. SoC/processor 130 may be a general-purpose processor, a digital signal processor (DSP), an application-specific integrated circuit (ASIC), a field-programmable gate array (FPGA) or other programmable logic device, discrete gate or transistor logic, discrete hardware components, or it may be a combination of these types of components.

[0028] Memory devices 170 may each include an array or arrays of memory cells to store digital information. Memory devices 170 may be configured to each operate with SoC/processor 130 and/or interface controller 120. In some examples, memory devices 170

may be configured to provide buffer memory for a memory bank for SoC/processor 130 or interface controller 120. Device 105 may include any number of memory devices 170.

[0029] Memory device 175 may include an array of memory cells and a local memory controller configured to operate with the array of memory cells. The array of memory cells included in memory device 175 may be structured in two or more tiers, which may each have different performance capabilities. The local memory controller of memory device 175 may also be configured to operate with SoC/processor 130 or interface controller 120. In some examples, first-tier memory cells may be 3D XPoint™ memory, which may provide a high number of input/output operations per second (IOPS) with a short response time to handle various workloads. In some examples, second-tier memory cells may be three-dimensional Not-AND (NAND) memory, which may provide high capacity for data storage at a relatively lower cost than the first-tier memory cells. The local memory controller of memory device 175 may be configured to facilitate the efficient operation of memory cells within memory device 175, which may have different characteristics among memory cells in the two or more tiers, with SoC/processor 130. Memory device 175 may include other types or combinations of memory arrays in some cases. In some examples, one or more memory devices 175 may be present in device 105.

[0030] Memory devices 180 may include one or more arrays of memory cells and a local memory controller configured to operate with the one or more arrays of memory cells. The local memory controller of a memory device 180 may also be configured to operate with SoC/processor 130 or interface controller 120. In some examples, a memory device 180 may include non-volatile or volatile memory cells, or a combination of both non-volatile and volatile memory cells. A non-volatile memory cell (e.g., an FeRAM memory cell) may maintain its stored logic state for an extended period of time in the absence of an external power source, thereby reducing or eliminating requirements to perform refresh operations (e.g., refresh operations such as those associated with DRAM cells).

[0031] The inclusion of an array of non-volatile memory cells (e.g., FeRAM memory cells) in a memory device 180 may provide various benefits (e.g., efficiency benefits) for device 105. Such benefits may include near-zero standby power (which may increase battery life), instant-on operation following a standby or un-powered (e.g., "off") state, and/or high areal memory density with low system power consumption relative to an array of volatile memory cells. Such features of non-volatile memory system or sub-system may, for example,

support the use of computationally intensive (e.g., desktop applications) operations or software in mobile environments. In some examples, one or more memory devices 180 may be present in device 105. In some cases, device 105 may include multiple kinds of non-volatile memory arrays employing different non-volatile memory technologies, such as one or more FeRAM arrays along with one or more non-volatile memory arrays using other memory technology. Further, the benefits described herein are merely exemplary, and one of ordinary skill in the art may appreciate further benefits.

[0032] In some cases, a memory device 180 may use a different page size than SoC/processor 130. In the context of a memory device, a page size may refer to a size of data handled at various interfaces, and different memory device types may have different page sizes. In some examples, SoC/processor 130 may use a DRAM page size (e.g., a page size in accord with one or more JEDEC low power double data rate (LPDDR) specifications), and a device 180 within device 105 may include an array of non-volatile memory cells that are configured to provide a different page size (e.g., a page size smaller than a typical DRAM page size). In some examples, a memory device 180 may support a variable page size—e.g., a memory device 180 may include an array of non-volatile memory cells (e.g., an FeRAM array) that supports multiple page sizes, and the page size used may vary from one access operation to another—and the local memory controller of memory device 180 may be configured to handle a variable page size for a memory array within memory device 180. For example, in some cases, a subset of non-volatile memory cells connected to an activated word line may be sensed simultaneously without having to sense all non-volatile memory cells connected to the activated word line, thereby supporting variable page-size operations within memory device 180. In some cases, the page size for an array of non-volatile memory cells may vary dynamically depending on the nature of an access command and a characteristic of (e.g., size or associated latency) associated data (e.g., data subject to the access command). Smaller page size may provide benefits (e.g., efficiency benefits) as a smaller number of memory cells may be activated in connection with a given access operation. The use of variable page size may provide further benefits to device 105, such as configurable and efficient energy usage when an operation is associated with a small change in information by reducing the page size while supporting a high-performance operation by increasing the page size when desired.

[0033] In some examples, device 105 may employ “prefetching” techniques to facilitate communication with SoC/processor 130. For instance, device 105 may cause a memory

device 180 to transmit data to memory device 175 that has not yet been requested by SoC/processor 130. In this way, SoC/processor 130 may access requested data located at memory device 175 using a first attempt instead of performing a delayed read operation to retrieve data from a memory device 180 using a second additional attempt. Moreover, SoC/processor 130 may access the data located in memory device 175 according to a memory access protocol configured at the SoC/processor 103.

[0034] Additionally or alternatively, device 105 may prefetch portions (e.g., subpages) of a memory page that includes data requested by SoC/processor 130. In this way, device 105 may further reduce power consumption and increase throughput at device 105. SoC/processor 130 may request data located in subpage of a memory page located at a memory device 180. The memory device 180 may transmit the subpage containing the requested data to SoC/processor 130 and/or memory device 175. In some examples, the memory device 180 may also transmit additional subpages in the memory page to memory device 175 together with (or, alternatively, in addition to) the subpage containing the requested data. In some cases, the decision to transmit additional subpages may be based on various factors or other indications (e.g., historical information related to access operations for the memory page). In some examples, a prefetch counter may be used to monitor past access operations performed on the subpages in the memory page. That is, a value of the prefetch counter may be updated (e.g., dynamically) based on prior and subsequent access operations performed on one or more subpages in the memory page. In some examples, each memory page may be associated with a respective prefetch counter.

[0035] DMAC 155 may support direct memory access (e.g., read or write) operations by SoC/processor 130 with respect to memory devices 170, 175, or 180. For example, DMAC 155 may support access by SoC/processor 130 of a memory device 170, 175, or 180 without the involvement or operation of interface controller 120.

[0036] Peripheral component(s) 150 may be any input or output device, or an interface for any such device, that may be integrated into device 105. Examples of peripheral component(s) 150 may include disk controllers, sound controllers, graphics controllers, Ethernet controllers, modems, universal serial bus (USB) controllers, serial or parallel ports, or peripheral card slots, such as peripheral component interconnect (PCI) or accelerated graphics port (AGP) slots. Peripheral component(s) 150 may also include other components understood by those skilled in the art as peripherals.

[0037] BIOS component 140 or board support package (BSP) 145 may be software components that include a basic input/output system (BIOS) operated as firmware, which may initialize and run various hardware components of system 100. BIOS component 140 or BSP 145 may also manage data flow between SoC/processor 130 and the various components, e.g., peripheral component(s) 150, input/output controller 135, etc. BIOS component 140 or BSP 145 may include a program or software stored in read-only memory (ROM), flash memory, or any other non-volatile memory. In some cases, BIOS component 140 and BSP 145 may be combined as a single component.

[0038] Input/output controller 135 may manage data communication between SoC/processor 130 and other devices, including peripheral component(s) 150, input devices 160, or output devices 165. Input/output controller 135 may also manage peripherals that are not integrated into device 105. In some cases, input/output controller 135 may include a physical connection or port to the external peripheral.

[0039] Input device 160 may represent a device or signal external to device 105 that provides input to device 105 or its components. In some cases, input device 160 may include a user interface or an interface with or between other devices (not shown in FIG. 1). In some cases, input device 160 may be a peripheral that interfaces with device 105 via peripheral component(s) 150 or is managed by input/output controller 135.

[0040] Output device 165 may represent a device or signal external to device 105 that is configured to receive output from device 105 or any of its components. For example, output device165 may include a display, audio speakers, a printing device, or another processor on printed circuit board, *etc.* In some cases, output device 165 may be a peripheral that interfaces with device 105 via peripheral component(s) 150 or is managed by input/output controller 135.

[0041] The components of device 105 may be made up of general purpose or specialized circuitry designed to carry out their respective functions. This may include various circuit elements, for example, conductive lines, transistors, capacitors, inductors, resistors, amplifiers, or other active or passive elements configured to carry out the functions described herein.

[0042] **FIG. 2** illustrates an exemplary memory system or sub-system that supports prefetch management for main memory in accordance with examples of the present disclosure. System 200 may include aspects of system 100 as described with reference to

FIG. 1 and may include a device 210. Device 210 may include aspects of device 105 as described with reference to FIG. 1. Device 210 may include memory sub-system 220, SoC/processor 250, and storage 260. SoC/processor 250 may be an example of an SoC/processor 130 as described with reference to FIG. 1. Memory sub-system 220 may be an example of a memory device 180 as described with reference to FIG. 1. Storage 260 may be an example of a memory device 175 as described with reference to FIG. 1.

[0043] SoC/processor 250 may be configured to operate with storage 260 via a bus 280 and with memory sub-system 220 via buses 270 and 275. In some examples, bus 280 may be configured to support periphery component interconnect express (PCIe) signaling. In other cases, bus 280 may employ another protocol, such as embedded Multi-media Control (eMMC), Universal Flash Storage (UFS), or the like. In some examples, bus 270 may be configured to support LPDDR command and address (CA) signaling, and bus 275 may be configured to support LPDDR input/output (I/O) signaling. In some examples, a local memory array may be disposed on a same substrate as SoC/processor 250 and may be configured to function as a cache memory 255 for SoC/processor 250.

[0044] Memory sub-system 220 may include non-volatile memory 225 and interface controller 230. Non-volatile memory 225 may be an example of a memory device 180 as described with reference to FIG. 1. Interface controller 230 may be an example of an interface controller 120 as described with reference to FIG. 1. Interface controller 230 may be configured to operate with SoC/processor 250 via buses 270 and 275 pursuant to an LPDDR specification (e.g., page size, timing requirements). Interface controller 230 may include virtual memory bank 235, which may be an example of a memory device 170 as described with reference to FIG. 1. In some examples, virtual memory bank 235 may include DRAM memory cells and may be configured to operate pursuant to an LPDDR specification. In some examples, virtual memory bank 235 may be disposed on a same substrate as interface controller 230. In addition, interface controller 230 may be configured to operate with non-volatile memory 225 via buses 271 and 276.

[0045] In some examples, memory sub-system 220 may further include buffer 240. In some examples, buffer 240 may include DRAM memory cells. Buffer 240 may be an example of a memory device 170 or a memory device 180 as described with reference to FIG. 1. In addition, interface controller 230 may be configured to operate with buffer 240 via buses 272 and 277. In some examples, bus 272 may be a buffer CA bus. In some examples,

bus 277 may be an interface (IF) buffer I/O bus. Interface controller 230 and buses 272 and 277 may be compatible with DRAM protocols. For example, interface controller 230 and buses 272 and 277 may utilize LPDDR page sizes and timings. In some examples, SoC/processor 250 may be configured to directly operate with buffer 240 via bus 275. In some examples, buffer 240 may be configured to have a page size compatible with bus 275, which may support direct access of buffer 240 by SoC/processor 250.

[0046] Buffer 240 may in some cases be configured to operate as a logical augmentation of cache memory 255 within SoC/processor 250. In some examples, the capacity of buffer 240 may be on the order of 256 M bytes. In some examples, the capacity of buffer 240 may be based at least in part on the size of cache memory 255 in SoC/processor 250. In some cases, buffer 240 may have a relatively small capacity, which may facilitate improved (e.g., faster) performance of memory sub-system 220 relative to a DRAM device of a larger capacity due to potentially smaller parasitic components, e.g., inductance associated with metal lines. A smaller capacity of buffer 240 may also provide benefits in terms of reducing system power consumption associated with periodic refreshing operations.

[0047] Memory sub-system 220 may be implemented in various configurations, including one-chip versions and multi-chip versions. A one-chip version may include interface controller 230, virtual memory bank 235, and non-volatile memory 225 on a single chip. In some examples, buffer 240 may also be included on the single-chip. In contrast, a multi-chip version may include one or more constituents of memory sub-system 220, including interface controller 230, virtual memory bank 235, non-volatile memory 225, and buffer 240, in a chip that is separate from a chip that includes one or more other constituents of memory sub-system 220. For example, in one multi-chip version, respective separate chips may include each of interface controller 230, virtual memory bank 235, and non-volatile memory 225. As another example, a multi-chip version may include one chip that includes both virtual memory bank 235 and interface controller 230 and a separate chip that includes buffer 240. Additionally, a separate chip may include non-volatile memory 225.

[0048] Another example of a multi-chip version may include one chip that includes both buffer 240 and virtual memory bank 235. Additionally, a separate chip may include both interface controller 230 and non-volatile memory 225 or respective separate chips may include each of interface controller 230 and non-volatile memory 225. In yet another example of a multi-chip version, a single chip may include non-volatile memory 225 and buffer 240.

Additionally, a separate chip may include both interface controller 230 and virtual memory bank 235 or respective separate chips may include each of interface controller 230 and virtual memory bank 235. In some examples, non-volatile memory 225 may include both an array of non-volatile memory cells and an array of DRAM cells. In some cases of a multi-chip version, interface controller 230, virtual memory bank 235, and buffer 240 may be disposed on a single chip and non-volatile memory 225 on a separate chip.

[0049] In some examples, non-volatile memory 225 may include an array of non-volatile memory cells (e.g., FeRAM memory cells) and, in some cases, may include a local memory controller (not shown). The non-volatile array included in non-volatile memory 225 may be configured to support fixed or variable page sizes (e.g., 64, 128, 192, or 256 bytes), which in some cases differ from a page size associated with SoC/processor 250. Further, interface controller 230, or in some cases a local memory controller included in non-volatile memory array 225, may be configured to determine a variable page size for non-volatile memory 225. Moreover, the non-volatile array may be configured with smaller page sizes (e.g., 64 bytes) than a DRAM array (2048 bytes), for example. In some examples, non-volatile memory 225 may be referred to as a non-volatile near memory to SoC/processor 250 (e.g., in comparison to storage 260). In the context of a memory system, a near memory may refer to a memory component placed near SoC/processor 250, logically and/or physically, to provide a faster access speed than other memory components. Configuring non-volatile memory 225 as a near memory for SoC/processor 250 may, for example, limit or avoid overhead that may be associated with SoC/processor 250 retrieving data from storage 260.

[0050] SoC/processor 250 may store critical information in non-volatile memory 225 upon unexpected power interruption instead of accessing storage 260, as accessing storage 260 may be associated with an undesired delay.

[0051] Interface controller 230 may be configured to operate with non-volatile memory 225 via buses 271 and 276. In some examples, bus 271 may be an FeRAM CA bus. In some examples, bus 276 may be an FeRAM interface (IF) bus. Interface controller 230 and buses 271 and 276 may be compatible with the page size of non-volatile memory 225. In some examples, buffer 240 may be configured to facilitate data transfer to non-volatile memory 225 via bus 295. In some examples, non-volatile memory 225 may be configured to facilitate data transfer to and from virtual memory bank 235 via bus 290.

[0052] Interface controller 230 may support low latency or reduced power operation (e.g., from the perspective of SoC/processor 250) by leveraging virtual memory bank 235 or buffer 240. For example, upon receiving a read command from SoC/processor 250, interface controller 230 may attempt to retrieve data from virtual memory bank 235 or buffer 240 for transmission to SoC/processor 250. If data subject to the read command is not present in virtual memory bank 235 or buffer 240, interface controller 230 may retrieve data from non-volatile memory 225 to store the data in virtual memory bank 235 and also (e.g., concurrently) send the data to SoC/processor 250.

[0053] Retrieving data from non-volatile memory 225 may be associated with increased latency compared to retrieving data from virtual memory bank 235 or buffer 240. In some examples, interface controller 230 may employ prefetching to proactively move data from non-volatile memory 225 to virtual memory bank 235. In this way, the memory access throughput of device 210 may be increased as the processor may access data stored in virtual memory bank directly, according to a DRAM memory access protocol.

[0054] Interface controller 230 may further reduce power consumption and latency by prefetching portions of a memory page. For example, interface controller 230 may receive a request for data from SoC/processor 250 that is located in a memory page of non-volatile memory 225. In some cases, the requested data may not be located in virtual memory bank 235 or buffer 240. Interface controller 230, or in some cases a local memory controller included in non-volatile memory array 225, may identify the memory page of non-volatile memory 225, and the portion, or subpage(s), of the memory page that includes the requested data. Non-volatile memory 225 may transmit the subpage including the requested data to virtual memory bank 235. In some cases, interface controller 230, or in some cases a local memory controller included in non-volatile memory array 225, may also identify a value of a prefetch counter associated with the memory page that includes the requested data, and non-volatile memory 225 may transmit additional subpages of the memory page with the subpage(s) that include the requested data based on the value of the prefetch counter. For example, for one value of the prefetch counter (e.g., logic 01), non-volatile memory 225 may transmit at least one additional subpage with the subpage(s) that include the requested data.

[0055] Interface controller 230 may manage operations of virtual memory bank 235. For example, interface controller 230 may use a set of flags located in virtual memory bank 235 to identify portions of virtual memory bank 235 storing valid data from non-volatile memory

225. As another example, upon receiving a write command from SoC/processor 250, interface controller 230 may store data at virtual memory bank 235. Another set of flags located in virtual memory bank 235 may indicate which portions of virtual memory bank 235 store valid data that are modified from corresponding contents of non-volatile memory 225. Valid data stored at virtual memory bank 235 may include data that have been retrieved from non-volatile memory 225 pursuant to a read command from SoC/processor 250 or data that have been received from SoC/processor 250 as a part of write command. Flags indicating which portions of virtual memory bank 235 store valid data or modified data may support interface controller 230 in saving only the data that has been modified from the corresponding contents in non-volatile memory 225. Furthermore, interface controller 230 may determine where to store data upon removal from virtual memory bank 235 (e.g., when SoC/processor 250 no longer needs the data). Interface controller 230 may monitor and identify the contents of virtual memory bank 235.

[0056] In some cases, interface controller 230 may include a counter that records a number (e.g., quantity or frequency) of access attempts by SoC/processor 250 to the contents of virtual memory bank 235 during a certain time interval. By way of example, if the counter shows that the number of access attempts by SoC/processor 250 during the time interval is less than a pre-determined threshold value, then upon removal of the data from virtual memory bank 235, interface controller 230 may store modified data (that is, data that was modified by the access attempts by SoC/processor 250) in non-volatile memory 225, as the interface controller 230 may anticipate, based on the relatively low number of prior access attempts, that SoC/processor 250 is not likely to access the data again for some duration of time. Or, if the counter indicates that the number of access attempts by SoC/processor 250 during the time interval is equal to or larger than the pre-determined threshold value, then interface controller 230 may, upon removal of the data from virtual memory bank 235, store the data in buffer 240, as the interface controller 230 may anticipate that SoC/processor 250 is likely to access the data soon. One skilled in the art may, in view of overall system requirements, devise various criteria (e.g., criteria including the threshold value of the counter, a clock, a value of the time interval, etc.) for interface controller 230 to use in making such determinations.

[0057] In some examples, interface controller 230 may record a number of access attempts by SoC/processor 250 of certain portions or subpages of a memory page stored at virtual memory bank. In some examples a portion of a memory page may include one or

more subpages of the memory page. Interface controller 230 may transmit the portions to the buffer based on the recorded number of access attempts for the individual portions. For example, interface controller 230 may transmit portions to buffer 240 that have been accessed at least a threshold number of times, and may discard or write-back to non-volatile memory 225 portions that have not been accessed the threshold number of times. A value of a prefetch counter for a memory page may be based on the portion of the memory page that is transmitted to buffer 240—e.g., may be based on a relationship between the portions sent to virtual memory bank 235 and the portion(s) of the memory page sent to buffer 240.

[0058] In addition, interface controller 230 may set up a by-pass indicator based on the counter when the number of access attempts by SoC/processor 250 is less than the pre-determined threshold value to by-pass saving the contents of virtual memory bank 235 to buffer 240. Then, interface controller 230 may directly save the modified contents of virtual memory bank 235 to non-volatile memory 225 based on the by-pass indicator. In some cases, upon removal of the data from virtual memory bank 235, interface controller 230 may determine that the data has not been modified since it was last retrieved from non-volatile memory 225 and may, based on that determination, discard the data (e.g., not write the data to either buffer 240 or non-volatile memory 225).

[0059] In some examples, a memory array in non-volatile memory 225 may be configured to store data in a first memory page configured with a first memory page size (e.g., 64, 128, 192, or 256 bytes). In some cases, the first memory page may be partitioned into a number of subpages. In some examples, the memory array may be configured with multiple memory pages that each share a common page size. For example, a 32 Gbyte memory array may include 500 million pages, each page having a page size of 64 bytes.

[0060] Virtual memory bank 235 may be configured to store data in a second memory page configured with a second memory page size (e.g., 2056 bytes) different from the first memory page size. In some examples, memory sub-system 220 may also include a prefetch counter associated with the first memory page. The prefetch counter may be configured to store a first value that corresponds to a first portion of the first memory page provided to the memory bank in response to a request for data located in the first memory page. In some cases, a prefetch counter may be included interface controller 230, or in some cases a local memory controller included in non-volatile memory array 225, or both. In some cases, a controller may include multiple prefetch counters. For example, a controller may include a

prefetch counter for each memory page, for each memory subpage in the memory array, or some combination.

[0061] Buffer 240 may be configured to receive data corresponding to a second portion of the first memory page. In some cases, a value of the prefetch counter may be based on the second portion of the first memory page—e.g., based on a relationship between the first portion and the second portion of the first memory page. In some cases, interface controller 230, or in some cases a local memory controller included in non-volatile memory array 225, may be configured to modify the value of the prefetch counter based on the relationship between the first portion and the second portion of the first memory page. Memory sub-system 220 may also include a second counter configured to monitor a number of access operations performed on all or a subpages of the first portion of the first memory page. In some cases, the data represented in the second portion of the first memory page transmitted to buffer 240 may be based on the monitored number of access operations. In some cases, the second counter may be included in interface controller 230, or in some cases a local memory controller included in non-volatile memory array 225.

[0062] In some cases, memory sub-system 220 may include means, such as interface controller 230, for receiving a first request for first data located in a memory page of non-volatile memory 225. Memory sub-system 220 may include means, such as interface controller 230, or in some cases a local memory controller included in non-volatile memory array 225, for identifying a first value of a prefetch counter associated with the memory page of the memory array. Memory sub-system 220 may include means, such as interface controller 230, or in some cases a local memory controller included in non-volatile memory array 225, for communicating, from the memory array to virtual memory bank 235, a first portion of the memory page based at least in part on the first value of the prefetch counter, the first portion of the memory page including the first data. Memory sub-system 220 may also include means, such as interface controller 230, for writing the first portion of the memory page to virtual memory bank 235.

[0063] Additionally, memory sub-system 220 may include means, such as interface controller 230, for communicating, to buffer 240, a second portion of the memory page, the second portion comprising at least a subset of the first data. Memory sub-system 220 may also include means, such as interface controller 230, or in some cases a local memory controller included in non-volatile memory array 225, for modifying the first value of the

prefetch counter based at least in part on a relationship between the first portion of the memory page and the second portion of the memory page. Memory sub-system 220 may include means, such as interface controller 230, or in some cases a local memory controller included in non-volatile memory array 225, for determining that the second portion of the memory page is different from the first portion of the memory page. And the means for modifying the first value of the prefetch counter may modify or adjust the first value of the prefetch counter based on determining that the second portion of the memory page is different from the first portion of the memory page.

[0064] Memory sub-system 220 may also include means, such as buffer 240, for evicting the second portion of the memory page, wherein modifying the first value of the prefetch counter is based at least in part on evicting the second portion of the memory page. Memory sub-system 220 may also include means, such as interface controller 230, for monitoring a number of access operations performed on the first portion of the memory page, where a second value of the prefetch counter is determined based at least in part on the number of access operations. In some examples, the means for monitoring the number of access operation monitors the number of access operations performed on one or more subsets, or subpages, of the first portion of the memory page.

[0065] Memory sub-system 220 may also include means, such as interface controller 230, for determining that a first number of access operations performed on a subset of the one or more subsets satisfies a threshold. And the means for communicating data to buffer 240 may communicate at least the subset to buffer 240 based at least in part on the determining. In some examples, the means for receiving the first request for first data further receives a second request for second data located in the memory page of non-volatile memory 225. Additionally, the means for communicating from non-volatile memory 225 to virtual memory bank 235 may also communicate a second portion of the memory page, wherein the first portion of the memory page is different from the second portion of the memory page, and wherein the second portion of the memory page comprises the second data. Moreover, the means for communicating from virtual memory bank 235 to buffer 240 may also communicate, from virtual memory bank 235 to buffer 240, a third portion of the memory page. The means for modifying a value of the prefetch counter may further modify the first value of the prefetch counter based on a relationship between the first portion of the memory page, the second portion of the memory page, or the third portion of the memory page, or any combination thereof.

[0066] The means for identifying a value of the prefetch counter may further identify a second value of the prefetch counter associated with the memory page, the second value being different from the first value. And the means for communicating from non-volatile memory 225 to virtual memory bank 235 may communicate a second portion of the memory page based at least in part on the second value of the prefetch counter, where the first portion of the memory page is different from the second portion of the memory page, and where the second portion of the memory page comprises the second data.

[0067] **FIG. 3** illustrates an exemplary data structure 300-a and state diagram 300-b that support prefetch management for main memory in accordance with examples of the present disclosure. Data structure 300-a illustrates a non-volatile memory page 310, a first field 315, and a second field 320.

[0068] Non-volatile memory page 310 may be used to store an amount of data. For instance, the non-volatile memory page 310 may be 128 or 256 bytes. Non-volatile memory page 310 may include a number of subpages, such as subpage 325, subpage 330, subpage 335, and subpage 340. Each subpage may be 64 bytes. In some examples, the subpages may be grouped as portions, such as portion 345.

[0069] Non-volatile memory page 310 may be associated with first field 315 and second field 320. In some examples, the first field 315 may be configured to indicate (and may be updated to track) a number of times a corresponding non-volatile memory page 310 has been accessed (e.g., read or write) by an SoC/processor (e.g., SoC/processor 250 described with reference to FIG. 2). In some examples, the first field 315 may be referred to as a saturating counter (SC). The first field 315 may in some cases include two bits of information, but it is to be understood that any number of bits may be used in accordance with the teachings herein. In some examples, non-volatile memory page 310 may be included in an FeRAM array.

[0070] In some examples, the second field 320 may be configured to indicate a size of data in a corresponding non-volatile memory page 310 to be retrieved upon receiving a read command. The size of data may be determined by an access pattern to the data made by an SoC/processor in one or more previous access operations and referred to as a prefetch size in some cases. A prefetch size may be an amount of data that is to be read in response to a read command for data included in the non-volatile memory page 310. For example, if data from the non-volatile memory page 310 is subject to a read command received by an interface

controller 230, or in some cases a local memory controller included in non-volatile memory array 225 as described in FIG. 2, (e.g., in anticipation of an access from an SoC/processor), the interface controller 230, or in some cases the local memory controller, may identify the subpage that includes the requested data and a value of the associated second field 320 to determine a prefetch size. The prefetch size may indicate a size of data (that includes and thus is at least as large as the requested data) to be read from the non-volatile memory 225 in response to the read request.

[0071] In some examples, logic states stored in the second field 320 may indicate a prefetch size of the corresponding non-volatile memory page 310. For example, "00" may correspond to 64 bytes, "01" may correspond to 128 bytes, "10" may correspond to 192 bytes, and "11" may correspond to 256 bytes. In such an example, if the interface controller 230, or in some cases a local memory controller included in non-volatile memory array 225, receives a read command for 64 bytes of data from a non-volatile memory page 310, and the associated second field 320 is 01, then the interface controller 230, or in some cases a local memory controller included in non-volatile memory array 225, may identify the prefetch size for the requested data as 192 bytes and read from the non-volatile memory 225 192 bytes of data, where the 192 bytes includes the requested 64 bytes. For example, the interface controller 230, or in some cases a local memory controller included in non-volatile memory array 225, may identify that the requested data is located in subpage 330 and may read portion 345. It is to be understood that the second field 320 may include any number of bits supporting any number of logic states and may indicate prefetch sizes of any size. In some examples, the second field 320 may be referred to as a prefetch (PF) counter.

[0072] In some examples, a subpage, such as subpage 340, may include a third field 350, which may be used to indicate (and may be updated to track) a number of times subpage 340 has been accessed. In some examples, the third field 350 may be referred to as a saturating counter (SC). The third field 350 may include two bits of information, but it is to be understood that any number of bits may be used in accordance with the present disclosure.

[0073] In some examples, an interface controller (e.g., interface controller 230 described with reference to FIG. 2, not shown in FIG. 3) may use a set of mode register bits to facilitate the SC and PF counter functionality of a non-volatile memory (e.g., non-volatile memory 225 described with reference to FIG. 2). In some examples, mode registers may establish various operation modes (e.g., different test modes, different read or write modes, different

performance modes) of a memory device and a set of bits associated with mode registers, which may be referred to as mode register bits, may be used to determine a particular mode of operation.

[0074] In some examples, an interface controller may access the contents of the SC and PF counter using a data mask inversion (DMI) pin along with data during a read operation. In some examples, an interface controller may write the contents of the SC and PF counter with a special command sequence. For example, an interface controller may provide the contents of SC and PF counter to registers associated with the SC and PF counter via column address pins during a write command issued to a non-volatile memory (e.g., non-volatile memory 225 described with reference to FIG. 2).

[0075] State diagram 300-b illustrates exemplary operational characteristics of a memory system or sub-system that support features and techniques as described herein. State diagram 300-b illustrates non-volatile memory 355, virtual page 385, and buffer 390. Non-volatile memory 355 may be an example of non-volatile memory 225 described with reference to FIG. 2. In some examples, non-volatile memory 355 includes non-volatile memory page 310.

[0076] Virtual page 385 may be a page within virtual memory bank 235 described with reference to FIG. 2. In some examples, virtual memory bank 235 may be a superset of multiple virtual pages 385. Buffer 390 may be an example of buffer 240 described with reference to FIG. 2. An interface controller (e.g., interface controller 230 described with reference to FIG. 2, not shown in FIG. 3) may perform or manage various operations (e.g., operations 360 through 380) associated with non-volatile memory 355, virtual page 385, and buffer 390. In some cases, an interface controller may manage an operation by requesting another entity (e.g., a local memory controller included in non-volatile memory array 225 described with reference to FIG. 2) to perform the operation.

[0077] Operation 360 may include transmitting the contents of a non-volatile memory page 310 from non-volatile memory 355 to virtual page 385 and storing the contents in virtual page 385. Operation 360 may be carried out by the interface controller in conjunction with a local memory controller (e.g., a local memory controller included in non-volatile memory array 225 as described with reference to FIG. 2) when an SoC/processor requests data corresponding to the contents of non-volatile memory page 310 that is not present either in the virtual page 385 or the buffer 390. Additionally, the interface controller may as part of operation 360 update a value of the first field 315 (e.g., a value of SC) associated with the

non-volatile memory page 310, to track a number of access events by the SoC/processor for the non-volatile memory page 310.

[0078] In some examples, operation 360 may include transmitting the contents of a non-volatile memory page 310 from non-volatile memory 355 to virtual page 385 based on or in response to receiving a request for certain data. In other examples, operation 360 may include transmitting the contents of a non-volatile memory page 310 from non-volatile memory 355 to virtual page 385 before receiving a request for certain data (e.g., during a prefetch operation). In some cases, operation 360 may include transmitting a portion of a non-volatile memory page 310 stored in non-volatile memory 355 to virtual page 385 before receiving a request for certain data or after receiving a request for certain data located in a subpage of non-volatile memory page 310 (e.g., during a partial prefetch operation).

[0079] For example, an interface controller 230, or in some cases a local memory controller included in non-volatile memory array 225, may receive a request for data located in subpage 335. The interface controller 230, or in some cases a local memory controller included in non-volatile memory array 225, may identify a value of the second field 320 based on receiving the request and may transmit a portion of non-volatile memory page 310 based on the value of the second field 320. For example, the interface controller 230, or in some cases a local memory controller included in non-volatile memory array 225, may determine that the value of the second field is logic 10, and may transmit portion 345, having a size of 192 bytes to virtual page 385. Additionally, the interface controller may as part of operation 360 update a value of a third field, such as third field 350, (e.g., a value of a respective SC) associated with the subpages 325, 330, and 335, to track a number of access events by the SoC/processor for the subpages 325, 330, and 335.

[0080] The interface controller may perform operation 365 when data requested by an SoC/processor (e.g., subject to a read command sent to the interface controller by the SoC/processor) is found in virtual page 385. As part of operation 365, the interface controller may retrieve the requested data from the virtual page 385 and provide the requested data to the SoC/processor without accessing either non-volatile memory 355 or buffer 390. In some examples, providing the requested data includes transmitting or accessing data corresponding to a full non-volatile memory page Additionally, the interface controller may update a value of the first field 315 (e.g., a value of SC) associated with the data, to track a number of access events by the SoC/processor for the non-volatile memory page 310.

[0081] In some examples, providing the requested data includes transmitting or accessing data corresponding to a portion of a non-volatile memory page. Additionally, the interface controller may update a value of a third field, such as third field 350, (e.g., a value of a respective SC) associated with the data, to track a number of access events by the SoC/processor for a subpage of non-volatile memory page 310.

[0082] The interface controller may perform operation 370 when a page in virtual page 385 is closed and a value of the first field 315 (e.g., a value of SC) associated with the closed page does not satisfy a threshold value. Virtual page 385 may include one or more pages within virtual memory bank 235 described with reference to FIG. 2. The interface controller may determine to close a page in virtual page 385 when the SoC/processor no longer needs the data associated with the page. Upon determining to close a page in virtual page 385, the interface controller may remove the data to make the memory space corresponding to the page available for the SoC/processor. Similarly, the interface controller may perform operation 370 when a subpage in virtual page 385 is closed and a value of a respective third field, such as third field 350 (e.g., a value of SC), associated with the closed page does not satisfy a threshold value.

[0083] In some cases, the interface controller may use a threshold value to determine how to dispose data from a closed page of virtual page 385. In some examples, when a value corresponding to first field 315 (e.g., a value of SC) is less than the threshold value, the interface controller may bypass saving data from a closed page to buffer 390. Instead, the interface controller may store any modified data from the closed page in non-volatile memory 355 and discard any unmodified data from the closed page. In such cases, the interface controller may determine whether data from a closed page includes a portion that the SoC/processor has modified relative to corresponding data stored in non-volatile memory 355. During operation 370, the interface controller may store any modified portion of the data of the closed page in non-volatile memory 355 from virtual page 385. Further, the interface controller may discard any unmodified data from a closed page after determining that the data has not been modified (that is, the interface controller may bypass storing an unmodified portion of the data in non-volatile memory 355). The interface controller may, in view of overall system requirements, determine the threshold value based on various criteria (e.g., a pre-determined value associated with a number of access to the page, a value of a time interval associated with lack of access to the page).

[0084] The interface controller may perform operation 375 when the interface controller determines to close a page in virtual page 385 and determines that a value of the first field 315 (e.g., a value of SC) associated with the closed page satisfies the threshold value described above. In some examples, when a value of the first field 315 (e.g., a value of SC) is equal to or greater than the threshold value, the interface controller may save data from a closed page to buffer 390, as the interface controller may determine that the SoC/processor is likely to access the data soon. As such, as a part of operation 375, the interface controller may store data from the closed page in buffer 390. Similarly, the interface controller may determine that a value of a third field (e.g., a value of SC) associated with a subpage of the closed page satisfies the threshold value described above. In some examples, when a respective value of the third field 350 (e.g., a value of SC) is equal to or greater than the threshold value, the interface controller may save data from a second portion of non-volatile memory page 310—e.g., the subpages of a closed page having a value of a third field that is equal to or greater than the threshold value—to buffer 390, as the interface controller may determine that the SoC/processor is likely to access those subpages soon. For example, the interface controller may save subpage 325 and subpage 335 to buffer 390.

[0085] The interface controller may perform operation 380 when it evicts a page from buffer 390. The interface controller may determine to evict a page from buffer 390 when the page is not accessed by the SoC/processor for a predetermined duration. In some cases, data from an evicted page may include a portion that has been modified by the SoC/processor relative to corresponding data stored in non-volatile memory 355. In such cases, as a part of operation 380, the interface controller may store only a modified portion of the evicted data in non-volatile memory 355. Additionally, as part of operation 380, the interface controller may update (e.g., reset to zero) a value of the first field 315 (e.g., a value of the SC) associated with the evicted page. Further, the interface controller may discard data after determining that the data has not been modified (that is, the interface controller may bypass storing an unmodified portion of the evicted data in non-volatile memory 355). Similarly, the interface controller may update (e.g., reset to zero) a value of a respective third field, such as third field 350, associated with an evicted subpage.

[0086] Additionally, as part of operation 380, the interface controller may update a value of the second field 320. For example, interface controller may decrease the value of the second field if a second portion of non-volatile memory page 310 stored to buffer 390 is

smaller in size than the first portion of non-volatile memory page 310 transmitted to virtual page 385.

[0087] **FIG. 4A** illustrates an exemplary process flow 400-a for a prefetch operation in accordance with examples of the present disclosure. Process flow 400-a depict aspects of a prefetch operation involving host 401, which may be an example of an SoC/processor as discussed with reference to FIG. 2, and memory sub-system 402 which may be an example of a memory sub-system as discussed with reference to FIG. 2. Memory sub-system 402 may include memory controller 403, which may be an example of an interface controller 230 or a local memory controller included in non-volatile memory array 225, or a combination thereof, as discussed with reference to FIG. 2; buffer 404, which may be an example of a buffer as discussed with reference to FIGs. 2 and 3; virtual memory bank 405, which may be an example of a virtual memory bank or virtual page as discussed with reference to FIGs. 2 and 3; and non-volatile memory 406, which may be an example of a non-volatile memory as discussed with reference to FIGs. 2 and 3. Memory controller 403, buffer 404, virtual memory bank 405, and non-volatile memory 406 may communicate with one another over one or more data busses.

[0088] At 410, host 401 may transmit an access request for data (e.g., to read or write data) stored in memory sub-system 402. In some cases, the access request may be transmitted according to a certain memory access protocol (e.g., a DRAM memory access protocol). In some examples, the access request may be sent directly to one or more components of memory sub-system 402. For example, the access request may be transmitted directly to buffer 404 and/or virtual memory bank 405, which may both operate according to the certain memory access protocol used by host 401. In some examples, the access request may be a request to read 64 bytes of data stored in memory sub-system 402. In other examples, the access request may be a request to read 128 bytes of data stored in memory sub-system 402, and so on. Additionally or alternatively, the access request may be sent indirectly to components of memory sub-system 402 via memory controller 403. For example, the access request may be sent to memory controller 403 which may relay the access request to buffer

[0089] At 415, memory controller 403 may access the requested data based on receiving the access request from host 401. Memory controller may forward the access request received from host 401 to one or more of buffer 404, virtual memory bank 405, and non-volatile memory 406. In some cases, memory controller may forward the access request to buffer

404, then to virtual memory bank 405, and last to non-volatile memory 406—e.g., after determining the requested data is not in buffer 404 or virtual memory bank 405. In some cases, memory controller 403 may forward the access request only to non-volatile memory 406—e.g., after determining the requested data is not in buffer 404 or virtual memory bank 405. In some examples, non-volatile memory 406 may include an FeRAM memory array.

[0090] At 420, buffer 404, memory controller 403 may determine that the requested data is not located in buffer 404 or virtual memory bank 405.

[0091] At 425, memory controller 403 may identify the non-volatile memory page (e.g., non-volatile memory page 310 of FIG. 3) located in non-volatile memory 406 that contains the requested data. The non-volatile memory page may be partitioned into multiple subpages. In some examples, memory controller 403 may also identify a subpage (e.g., subpage 330 of FIG. 3) of the non-volatile memory page that contains the requested data. In some cases, the subpage that contains the requested data may be 64 bytes. In other examples, the size of the requested data may be 128 bytes, and two subpages may contain the requested data.

[0092] Additionally, memory controller 403 may identify a value of a prefetch counter (e.g., second field 320 of FIG. 3) associated with the non-volatile memory page. In some cases, the prefetch counter may be initialized with a value of logic 00. In other cases, the prefetch counter may be initialized with a value of logic 11. Initializing a value of the prefetch counter with logic 11 may cause non-volatile memory 406 to initially transmit more data (e.g., at power-on), increasing power consumption and throughput. Conversely, a value of the prefetch counter with logic 00 may cause non-volatile memory 406 to initially transmit less data, decreasing power and potentially lowering throughput. In some examples, the selected initialization value corresponds to a power mode (e.g., high low power mode) or operation mode (e.g., high throughput mode) of the device.

[0093] At 430, memory controller 403 may identify a first portion (e.g., portion 345 of FIG 3.) of the non-volatile memory page containing the requested data based on the value of the prefetch counter. The first portion may include at least the subpage that contains the requested data. In some examples, a size of the first portion may be based on the value of the prefetch counter. For example, if the value of the prefetch counter equals logic 00, then a size of the first portion may be equal to 64 bytes (e.g., the first portion may only include the subpage containing the data). In another example, if the value of the prefetch counter equals logic 01, then a size of the first portion may be equal to 128 bytes (e.g., the first portion may

include the subpage containing the requested data as well as a second subpage). In another example, if the value of the prefetch counter equals logic 10, then a size of the first portion may be equal to 192 bytes (e.g., the first portion may include the subpage containing the requested data, a second subpage, and third subpage), and so on.

[0094] In some examples, the size of the requested data may override the prefetch counter value. For example, if the size of the requested data is 128 bytes, and the prefetch counter value corresponds to size of 64 bytes, then a size of the first portion may be equal to 128 bytes (e.g., the first portion may include the two subpages containing the requested data). Other mappings between a value of a prefetch counter and the size of the first portion may be alternatively used and are contemplated based on the present disclosure. For instance, a prefetch value of logic 00 may correspond to 128 bytes, a prefetch value of logic 01 may correspond to 256 bytes, a prefetch value of logic 10 may correspond to 384 bytes, and so on.

[0095] In some examples, memory controller 403 may select which subpages to include with the subpage containing the requested data based on one or more factors or indicators (e.g., historical access operations performed on the subpages in the non-volatile memory page). For example, memory controller 403 may identify that subpages of a memory page, other than a subpage of the memory page containing the requested data, were previously accessed in virtual memory bank 405, and may transmit the identified subpages with the subpage containing the requested data in subsequent read operations. In this way, memory controller 403 may anticipate subsequent data requests from host 401 and avoid delays associated with accessing non-volatile memory 406.

[0096] For instance, memory controller 403 may monitor the data stored in virtual memory bank 405 and identify which subpages correspond to data accessed in virtual memory bank 405. In some cases, memory controller 403 may identify that data corresponding to a subpage has been accessed more than a threshold number of time (e.g., more than once) if the data may be moved to buffer 404, and may transmit, in a subsequent access request for data in the non-volatile memory page, the identified subpages with the subpage containing the data requested in the subsequent access request. In some examples, memory controller 403 may track a number of access operations performed on data corresponding to each subpage of the non-volatile memory page and may transmit the subpage with the most access attempts within a certain period of time with the subpage

containing the requested data. A respective saturation counter may be used to track a number of access operations performed on each subpage of a non-volatile memory page.

[0097] Additionally, memory controller 403 may keep track of when access operations are performed on subpages of the non-volatile memory page. For example, memory controller 403 may choose to include the most recently accessed subpages in subsequent transmissions of data from the non-volatile memory page. Additionally, memory controller 403 may identify that two subpages are often accessed together, set the prefetch value to logic 01, and transmit the two subpages together when data in one of the subpages is accessed.

[0098] At 435, memory sub-system 402 may transmit, or memory controller 403 may cause non-volatile memory 406 to transmit, data corresponding to the first portion of the non-volatile memory page to host 401. Accordingly, virtual memory bank 405 may store data corresponding to the first portion of the non-volatile memory page. In some examples, storing data corresponding to the first portion of the non-volatile memory page includes storing data corresponding to respective saturation counters for each of the subpages.

[0099] At 440, memory sub-system 402 may transmit, or memory controller 403 may cause non-volatile memory 406 to transmit, data corresponding to the first portion of the non-volatile memory page to virtual memory bank 405.

[0100] In some examples, 410, 415, and 420 may be omitted from process flow 400-a. For example, memory controller 403 may preemptively move non-volatile memory page and/or non-volatile memory sub-pages to virtual memory bank 405. In some cases, memory controller 403 may move a portion of a non-volatile memory page to virtual memory bank 405 based on a value of a prefetch counter. That is, memory controller 403 may identify a subpage of a non-volatile memory page to be moved to virtual memory bank and may move additional subpages of the non-volatile memory page to virtual memory bank based on a value of a prefetch counter associated with the non-volatile memory page.

[0101] **FIG. 4B** illustrates an exemplary process flow 400-b for a prefetch operation in accordance with examples of the present disclosure. Process flow 400-b depict aspects of a prefetch operation involving host 401 and memory sub-system 402, which may include memory controller 403, buffer 404, virtual memory bank 405, and non-volatile memory 406. The operations performed in process flow 400-b may be a follow on to the operations performed in process flow 400-a as discussed with reference to FIG. 4A.

[0102] At 445, memory sub-system 402 may monitor access operations performed on the first portion of the non-volatile memory page. For example, memory sub-system 402 may track a respective number of access operations performed on each subpage of the first portion of the non-volatile memory page (e.g., using a respective saturation counter). Virtual memory bank 405 may also keep track of which data stored in the subpages is valid—e.g., which data stored in virtual memory bank 405 matches (e.g., has not been overwritten) the data stored in non-volatile memory 406.

[0103] At 450, host 401 may transmit a number of access requests for data stored in memory sub-system 402. As above, the access request may be transmitted according to a certain memory access protocol, and may be transmitted either directly or indirectly to the components of memory sub-system 402.

[0104] At 455, memory controller 403 may forward the access requests to one or more of buffer 404, virtual memory bank 405, or non-volatile memory 406. In some examples, an access request may request data that is not located at buffer 404 or at virtual memory bank 405 (which may be referred to as a "miss"). After a miss, memory controller 403 may inform host 401 of the miss and indicate that there will be a delay in communicating the requested data. Memory controller 403 may then access the non-volatile memory page and may transmit, or cause non-volatile memory 406, to transmit a portion or subpage of the non-volatile memory containing the requested data to virtual memory bank 405 and/or host 401.

[0105] At 460, memory sub-system 402 may identify a second portion of the non-volatile memory page. In some examples, data corresponding to the second portion of the non-volatile memory page may be equivalent to or overlap with the data corresponding to the first portion of the non-volatile memory page. In other examples, data corresponding to the second portion of the non-volatile memory page may be different than, or non-overlapping with, the data corresponding to the first portion of the non-volatile memory page.

[0106] For example, the first portion of the non-volatile memory page may include data corresponding to a first, second, and third subpage, while the second portion of the non-volatile memory page may include data corresponding to the first and third subpages. In some examples, memory sub-system 402 determines which data corresponding to which subpages to include in the second portion based on the monitored access operations. For example, memory sub-system 402 may select the first and third subpages for inclusion in the second portion based on determining that a respective number of access operations had been

performed on each of the two subpages (e.g., more than a threshold value), and may exclude the second subpage from inclusion in the second portion based on determining that a respective number of access operations were performed on the second subpage (e.g., less than a threshold value). In some instances, the second subpage may contain the data originally requested by host 401 at step 410.

[0107] At 465, memory sub-system 402 may transmit data corresponding to the second portion of the non-volatile memory page to buffer 404. In some examples, memory sub-system 402 may transmit the data to buffer 404 after closing a virtual page that includes the first portion of the non-volatile memory page at virtual memory bank 405—e.g., when SoC/processor no longer needs access to the virtual page or when the virtual page has not been accessed for a certain period of time. Accordingly, buffer 404 may store the data corresponding to the second portion of the non-volatile memory page.

[0108] At 470, buffer 404 may evict the second portion of the non-volatile memory page. For example, buffer 404 may evict the data corresponding to the second portion of the non-volatile memory page after a certain time period of time has passed or after the data has not been access for a certain period of time. Buffer 404 may also write back data that has been modified to non-volatile memory 406. Additionally, a saturation counter associated with an evicted page or subpage may be reset.

[0109] At 475 memory sub-system 402 may update the prefetch counter associated with the non-volatile memory page identified at 425 of process flow 400-a. The update to the prefetch counter may be based on a relationship between the evicted data corresponding to the second portion of the memory page and the data corresponding to the first portion of the memory page. For example, memory sub-system 402 may determine that a size of the data corresponding to the second portion of the non-volatile memory page is smaller than a size of the data corresponding to the first portion of the non-volatile memory page, and may decrease the value of the prefetch counter. In some examples, memory sub-system 402 may determine that a size of the data corresponding to the second portion of the non-volatile memory page is larger than a size of the data corresponding to the first portion of the non-volatile memory page—e.g., when additional data in the memory page, but external to the first portion of the memory page, is requested by host 401—and may increase the value of the prefetch counter.

[0110] In some cases, memory sub-system 402 may select a value of the prefetch counter to correspond to the size of the second portion evicted from the buffer. For example, if a size

of the evicted second portion is 256 bytes, then a value of the prefetch counter may be updated to logic 11. In other examples, memory sub-system 402 may determine that the subpages corresponding to the data written to buffer 404 are different than one or more of the subpages corresponding to the data written to virtual memory bank 405, and may modify a value of the prefetch counter. In some examples, memory sub-system 402 may update the prefetch counter prior to 475. For example, memory sub-system 402 may similarly update a value of the prefetch counter after 465.

[0111] After 475, process flow 400-b may return to 410 or 425 of process flow 400-a. In some examples, host 401 transmits a request for data located in the same non-volatile memory page as above. As previously discussed, it may be determined that the requested data is not located in buffer 404 or virtual memory bank 405, and memory sub-system 402 may identify the same non-volatile memory page as containing the requested data. Also as previously discussed, the memory sub-system 402 may identify a third portion of the memory page based on a value of the prefetch counter. In some examples, the identified third portion may be different than the first identified portion based on an updated value of the prefetch counter. For example, the first identified portion may correspond to the first, second, third subpages based on the first subpage containing the requested data and the prefetch counter having a value of logic 10. While the third portion may correspond to the first and third subpages, based on the first subpage containing the requested data and the prefetch counter having an update value of logic 01—e.g., based on memory sub-system identifying that the second subpage was moved to buffer 404, while the third subpage was not.

[0112] **FIG. 5** shows a block diagram 500 of a memory controller 505 that supports prefetch management for main memory in accordance with examples of the present disclosure. Memory controller 505 includes memory interface component 510, prefetch component 515, data management component 520, and access management component 525. Each of these components may communicate, directly or indirectly, with one another (e.g., via one or more buses). In some examples, memory controller 505 manages operations across a memory sub-system. That is, memory controller 505 may manage or facilitate communications between a memory array, a virtual memory bank, and a buffer. For instance, memory controller 505 may cause certain components of a memory sub-system to write or read data.

[0113] Memory interface component 510 may manage commands received from an external processor or SoC. For instance, memory interface component 510 may receive, from a processor, a request for data located in a memory page of a memory array that is coupled with memory interface component 510.

[0114] Prefetch component 515 may manage prefetch operations based on a received request for data located in the memory page of the memory array. For instance, prefetch component 515 may include a prefetch counter 530 and may identify a value of the prefetch counter 530 associated with the memory page containing the requested data.

[0115] Data management component 520 may communicate, from the memory array to a memory bank, a portion of the memory page based on the identified value of the prefetch counter 530, where the portion includes the requested data. In some examples, data management component 520 writes the portion of the memory page to the memory bank.

[0116] Access management component 525 may monitor a number of access operations performed on the portion of the memory page while stored at the memory bank. In some examples, access management component 525 tracks a number of times the memory page and/or portions of the memory page are accessed. In some instances, access management component 525 includes one or more counters 535 (e.g., saturation counters) to keep track of the number of access operations performed on the memory page. A second value of the prefetch counter 530 may be determined based on the monitored number of access operations. For example, access management component 525 may monitor the number of access operations performed on a subset, or subpage, of the portion of the memory page. Access management component 525 may determine that a number of access operations performed on a subpage satisfies a threshold—e.g., a value of a respective counter 535 is greater than a threshold value—and may communicate data corresponding to the subpage to a buffer based on the determination.

[0117] In some examples, data management component 520 may also communicate to a buffer another portion of the memory page. In some examples, the other portion includes the requested data. In some cases, prefetch component 515 modifies a value of the prefetch counter 530 based on a relationship between the first portion of the memory page and the second portion of the memory page. For example, prefetch component 515 may determine that the data represented by the second portion of the memory page is different than the data represented by the first portion of the memory page, and may modify, or adjust (e.g.,

increment/decrement) the value of the prefetch counter 530 based on determining that the portions are different.

[0118] Data management component 520 may also cause the buffer to evict the other portion of the memory page. In some cases, the prefetch component 515 modifies the value of the prefetch counter 530 based on the other portion being evicted from the buffer—e.g., in response to the eviction.

[0119] In some examples, memory interface component 510 may receive a second request for data located in the memory page of the memory array and may determine that the requested data is not located in the virtual memory bank or the buffer. Accordingly, data management component 520 may access the memory array and communicate a second portion including the second data of the memory page to the memory bank, where the first portion of the memory page (e.g., which may include subpage 1 and subpage 3) is different from the second portion of the memory page (e.g.., which may include subpage 2), and where the second portion of the memory page comprises the second data.

[0120] Data management component 520 may later communicate data corresponding to a third portion of the memory page (e.g., which may include subpage 1, subpage 2, and subpage 3) to the buffer, and prefetch component 515 may modify the value of the prefetch counter 530 based on a relationship between the original, first portion of the memory page communicated to the memory bank, the second portion of the memory page communicated to the memory bank, and the third portion of the memory page communicated to the buffer. For example, prefetch component 515 may increment the value of the prefetch counter 530 based on determining that data corresponding to three subpages was evicted from the buffer, while two subpages were transmitted to the virtual memory bank.

[0121] After the prefetch counter 530 is updated, memory interface component may receive another request for data located in the memory page of the memory array. In some cases, the first and second data are the same, while in other cases the first and second data are different. Prefetch component 515 may identify a second value (e.g., the updated value) of the prefetch counter 530 associated with the memory page, and data management component 520 may communication, from the memory array to the memory bank another portion of the memory page based on the updated value of the prefetch counter 530. In some cases, the other portion includes the data requested in the latest request, the other portion is different than the original, first portion transmitted to the memory bank.

[0122] **FIG. 6** shows a flowchart illustrating a method 600 for prefetch management for main memory in accordance with examples of the present disclosure. The operations of method 600 may be implemented by a memory system, sub-system, or its components as described herein. For example, the operations of method 600 may be performed by an memory controller, such as interface controller 230, as described with reference to FIG. 2, and may execute a set of codes to control the functional elements of the device to perform the functions below. Additionally or alternatively, a memory controller may perform aspects of the functions described below using special-purpose hardware.

[0123] At block 605, the memory controller may receive a first request for first data located in a memory page of a memory array of a device. The operations of block 605 may be performed according to the methods described with reference to FIGs. 3-4. In certain examples, some or all of the operations of block 605 may be performed by a memory interface component as described with reference to FIG. 5.

[0124] At block 610, the memory controller may identify a first value of a prefetch counter associated with the memory page of the memory array. The operations of block 610 may be performed according to the methods described with reference to FIGs. 3-4. In certain examples, some or all of the operations of block 610 may be performed by a prefetch component as described with reference to FIG. 5.

[0125] At block 615, the memory controller may communicate, from the memory array to a memory bank of the device, a first portion of the memory page based at least in part on the first value of the prefetch counter, the first portion of the memory page including the first data. The operations of block 615 may be performed according to the methods described with reference to FIGs. 3-4. In certain examples, some or all of the operations of block 615 may be performed by a data management component as described with reference to FIG. 5.

[0126] An apparatus is described. In some examples, the apparatus may include means for receiving a first request for first data located in a memory page of a memory array of a device, means for identifying a first value of a prefetch counter associated with the memory page of the memory array, and means for communicating, from the memory array to a memory bank of the device, a first portion of the memory page based at least in part on the first value of the prefetch counter, the first portion of the memory page comprising the first data.

[0127] In some examples the apparatus may include means for communicating, to a buffer of the device, a second portion of the memory page, the second portion comprising at least a subset of the first data and means for modifying the first value of the prefetch counter based at least in part on a relationship between the first portion of the memory page and the second portion of the memory page. In some examples the apparatus may include means for determining that the second portion of the memory page is different from the first portion of the memory page, wherein the first value of the prefetch counter is modified based at least in part on determining that the second portion of the memory page is different from the first portion of the memory page.

[0128] In some examples the apparatus may include means for evicting, by the buffer, the second portion of the memory page, wherein the first value of the prefetch counter is modified based at least in part on evicting the second portion of the memory page. In some examples, modifying the first value of the prefetch counter may include adjusting the first value of the prefetch counter based at least in part on determining that the first portion of the memory page is different than the second portion of the memory page.

[0129] In some examples the apparatus may include means for writing the first portion of the memory page to the memory bank and means for monitoring a number of access operations performed on the first portion of the memory page, wherein a second value of the prefetch counter is determined based at least in part on the number of access operations. In some examples, monitoring the number of access operations performed may include monitoring the number of access operations performed on one or more subsets of the first portion of the memory page.

[0130] In some examples the apparatus may include means for determining that a first number of access operations performed on a subset of the one or more subsets satisfies a threshold and means for communicating at least the subset to a buffer of the device based at least in part on the determining. In some examples the apparatus may include means for receiving a second request for second data located in the memory page of the memory array and means for communicating, from the memory array to the memory bank, a second portion of the memory page, wherein the first portion of the memory page is different from the second portion of the memory page, and wherein the second portion of the memory page comprises the second data.

[0131] In some examples the apparatus may include means for communicating, from the memory bank to a buffer of the device, a third portion of the memory page and means for modifying the first value of the prefetch counter based at least in part on a relationship between the first portion of the memory page, the second portion of the memory page, or the third portion of the memory page, or any combination thereof. In some examples the apparatus may include means for receiving a second request for second data located in the memory page of the memory array, means for identifying a second value of the prefetch counter associated with the memory page, wherein the second value is different from the first value, and means for communicating, from the memory array to the memory bank, a second portion of the memory page based at least in part on the second value of the prefetch counter, wherein the first portion of the memory page is different from the second portion of the memory page, and wherein the second portion of the memory page comprises the second data.

[0132] **FIG. 7** shows a flowchart illustrating a method 700 for prefetch management for main memory in accordance with examples of the present disclosure. The operations of method 700 may be implemented by a memory system, sub-system, or its components as described herein. For example, the operations of method 700 may be performed by an memory controller, such as interface controller 230, as described with reference to FIG. 2, and may execute a set of codes to control the functional elements of the device to perform the functions below. Additionally or alternatively, a memory controller may perform aspects of the functions described below using special-purpose hardware.

[0133] At block 705, the memory controller may receive a first request for first data located in a memory page of a memory array of a device. The operations of block 705 may be performed according to the methods described with reference to FIGs. 3-4. In certain examples, some or all of the operations of block 705 may be performed by a memory interface component as described with reference to FIG. 5.

[0134] At block 710, the memory controller may identify a first value of a prefetch counter associated with the memory page of the memory array. The operations of block 710 may be performed according to the methods described with reference to FIGs. 3-4. In certain examples, some or all of the operations of block 710 may be performed by a prefetch component as described with reference to FIG. 5.

[0135] At block 715, the memory controller may communicate, from the memory array to a memory bank of the device, a first portion of the memory page based at least in part on the first value of the prefetch counter, the first portion of the memory page including the first data. The operations of block 715 may be performed according to the methods described with reference to FIGs. 3-4. In certain examples, some or all of the operations of block 715 may be performed by a data management component as described with reference to FIG. 5.

[0136] At block 720, the memory controller may communicate, to a buffer of the device, a second portion of the memory page, the second portion comprising at least a subset of the first data. The operations of block 720 may be performed according to the methods described with reference to FIGs. 3-4. In certain examples, some or all of the operations of block 720 may be performed by a data management component as described with reference to FIG. 5.

[0137] At block 725, the memory controller may modify the first value of the prefetch counter based at least in part on a relationship between the first portion of the memory page and the second portion of the memory page. The operations of block 725 may be performed according to the methods described with reference to FIGs. 3-4. In certain examples, some or all of the operations of block 725 may be performed by a prefetch component as described with reference to FIG. 5.

[0138] **FIG. 8** shows a flowchart illustrating a method 800 for prefetch management for main memory in accordance with examples of the present disclosure. The operations of method 800 may be implemented by a memory system, sub-system, or its components as described herein. For example, the operations of method 800 may be performed by an interface controller, such as interface controller 230, as described with reference to FIG. 2, which may execute a set of codes to control the functional elements of the device to perform the functions below. Additionally or alternatively, the interface controller 230 may perform aspects of the functions described below using special-purpose hardware.

[0139] At block 805, the memory controller may receive a first request for first data located in a memory page of a memory array of a device. The operations of block 805 may be performed according to the methods described with reference to FIGs. 3-4. In certain examples, some or all of the operations of block 805 may be performed by a memory interface component as described with reference to FIG. 5.

[0140] At block 810, the memory controller may identify a first value of a prefetch counter associated with the memory page of the memory array. The operations of block 810

may be performed according to the methods described with reference to FIGs. 3-4. In certain examples, some or all of the operations of block 810 may be performed by a prefetch component as described with reference to FIG. 5.

[0141] At block 815, the memory controller may communicate, from the memory array to a memory bank of the device, data corresponding to a first portion of the memory page based at least in part on the first value of the prefetch counter, the first portion of the memory page including the first data. The operations of block 815 may be performed according to the methods described with reference to FIGs. 3-4. In certain examples, some or all of the operations of block 815 may be performed by a data management component as described with reference to FIG. 5. In some cases, the memory controller writes the first portion of the memory page, or data corresponding to the first portion of the memory page, to the memory bank.

[0142] In some cases, the memory controller may communicate, from the memory bank to a buffer of the device, data corresponding to a second portion of the memory array. In some cases, the second portion of the memory array may include the first requested data. The first portion of the memory array may be different than the second portion of the memory array—e.g., the first portion may include three subpages of the memory page, while the second portion of the memory array may include two subpages of the memory page. In some cases, the first portion may include three subpages of the memory page, while the second portion of the memory array may include four subpages of the memory page (e.g., in the event of a subsequent "missed" read).

[0143] The memory controller may modify the first value of the prefetch counter based on a relationship between the first portion of the memory page and the second portion of the memory page. For example, the memory controller may determine that the second portion of the memory page is different from the first portion of the memory page, and may modify the first value of the prefetch counter based on determining that the second portion of the memory page is different from the first portion of the memory page. In some cases, the memory controller, causes the buffer to evict the second portion of the memory page (e.g., after a certain time period has elapsed). The memory controller may modify the value of the prefetch counter based on the eviction of the second portion of the memory page. For example, the memory controller may wait to update the prefetch counter until the after the eviction occurs.

[0144] After the eviction, the memory controller may adjust the first value of the prefetch counter based on determining that the data corresponding to the first portion of the memory page and the data corresponding to the second portion of the memory page are different. For example, the memory controller may increment the first value of the prefetch counter based on determining that the first portion of the memory page corresponds to fewer subpages than the second portion of the memory page (e.g., the first portion corresponds to three subpages and the second portion corresponds to four subpages). In another example, the memory controller may decrement the value of the prefetch counter based on determining that the first portion of the memory page corresponds to more subpages than the second portion of the memory page. In yet another example, may increment the first value of the prefetch counter based on determining that the first portion of the memory page corresponds to at least one different subpage than the second portion of the memory page.

[0145] The memory controller may also monitor a number of access operations performed on the first portion of the memory page, and a second value of the prefetch counter may be based on the monitored number of access operations. For example, the memory controller may monitor a number of access operations performed on each subset, or subpage, of the first portion of the memory page and the number of subsets, or subpages, of the second portion of the memory page may be based on the monitored number of access operations. For instance, the memory controller may choose to include two of three subpages in the second portion of the memory page based on determining that the two subpages included in the second potion were accessed more than a threshold number of time, while the other subpage was accessed less than the threshold number of times.

[0146] In some cases, the memory controller may determine that a second request to access certain data in the virtual memory bank or buffer has "missed." Accordingly, memory controller may locate the requested data in the memory array and move the data to the virtual memory bank. In some examples, the second requested data is located in the same memory page as the first requested data, and the memory controller communicates a second portion of the memory page to the virtual memory bank, where the second portion of the memory page is different than the first portion of the memory page and includes the second requested data.

[0147] In such a case, the memory controller may communicate data corresponding to a third portion, including data corresponding to subpages from the first portion and subpages from the second portion of the memory page, to the buffer. After the buffer evicts the third

portion of the memory page, the memory controller may modify the first value of the memory controller based on a relationship between the first portion, the second portion, and the third portion of the memory array. For instance, if the first portion corresponds to a first set of two subpages, the second portion correspond to a second set of two subpages, and the third portion corresponds to all four subpages, the memory controller may increment the value of the prefetch counter.

[0148] At block 820 and after the prefetch counter has been updated to the second value, the memory controller may receive a second request for second data located in the memory page of the memory array. The operations of block 820 may be performed according to the methods described with reference to FIGs. 3-4. In certain examples, some or all of the operations of block 820 may be performed by a memory interface component as described with reference to FIG. 5. In some cases, the second data may be the same as the first data.

[0149] At block 825, the memory controller may identify a second value of the prefetch counter associated with the memory page, the second value being different from the first value. The operations of block 825 may be performed according to the methods described with reference to FIGs. 3-4. In certain examples, some or all of the operations of block 825 may be performed by a prefetch component as described with reference to FIG. 5.

[0150] At block 830, the memory controller may communicate, from the memory array to the memory bank, a second portion of the memory page based at least in part on the second value of the prefetch counter, wherein the first portion of the memory page is different from the second portion of the memory page, and wherein the second portion of the memory page comprises the second data. The operations of block 830 may be performed according to the methods described with reference to FIGs. 3-4. In certain examples, some or all of the operations of block 830 may be performed by a data management component as described with reference to FIG. 5.

[0151] An apparatus is described. In some examples, the apparatus may include a memory array configured to store data in a first memory page configured with a first memory page size, a memory bank configured to store data in a second memory page configured with a second memory page size different from the first memory page size, and a prefetch counter associated with the first memory page, the prefetch counter configured to store a first value that corresponds to a first portion of the first memory page provided to the memory bank in response to a request for data located in the first memory page.

[0152] In some examples, the apparatus may include a buffer configured to receive a second portion of the first memory page, wherein the first value of the prefetch counter is based at least in part on the second portion of the first memory page. In some examples, the apparatus may include a controller configured to modify the first value of the prefetch counter based at least in part on a relationship between the first portion of the first memory page and the second portion of the first memory page.

[0153] In some examples, the apparatus may include a second counter configured to monitor a number of access operations performed on the first portion of the first memory page, wherein the second portion of the first memory page is based at least in part on a second value of the second counter. In some examples, the apparatus may include a controller configured to monitor access operations performed on the first portion of the first memory page. In some examples, the first memory page is partitioned into a plurality of subsets of the first memory page.

[0154] In some examples, the memory array is configured to store a plurality of memory pages each configured with the first memory page size. In some examples, the apparatus may include a plurality of prefetch counters each associated with a corresponding memory page of the plurality of memory pages, wherein each of the plurality of prefetch counters is configured to store a respective value corresponding to a respective portion of a respective memory page, and wherein the respective portion is responsive to a respective request for data located in the respective memory page.

[0155] An apparatus is described. In some examples, the apparatus may include a memory array configured to store data in a first memory page configured with a first memory page size, a memory bank configured to store data in a second memory page configured with a second memory page size different from the first memory page size, a prefetch counter, wherein a first value of the prefetch counter is associated with the first memory page, and a controller coupled with the memory array, the memory bank, and configured to interface with a host. In some examples, the controller may be operable to receive a first request for first data located in the first memory page, identify the first value of the prefetch counter, and communicate, from the memory array to the memory bank, a first portion of the first memory page based at least in part on the first value of the prefetch counter, the first portion of the first memory page comprising the first data.

[0156] In some examples, the apparatus may include a buffer and the controller may be operable to communicate, from the memory bank to the buffer, a second portion of the first memory page. In some examples, the controller may be operable to modify the first value of the prefetch counter based at least in part on a relationship between the first portion of the first memory page and the second portion of the first memory page. In some examples, the controller may be operable to determine that the second portion of the first memory page is different from the first portion of the first memory page, wherein modifying the first value of the prefetch counter is based at least in part on determining that the second portion of the first memory page is different from the first portion of the first memory page.

[0157] In some examples, the controller may be operable to adjust the first value of the prefetch counter based at least in part on determining that the first portion of the first memory page is different than the second portion of the first memory page. In some examples, the controller may be operable to monitor a number of access operations performed on the first portion of the first memory page, wherein a second value of the prefetch counter is determined based at least in part on the number of access operations. In some examples, the controller may be operable to monitor the number of access operations performed on one or more subsets of the first portion of the first memory page.

[0158] An apparatus is described. In some examples, the apparatus may include a memory array configured according to a first memory page size, a memory bank configured according to a second memory page size different than the first memory page size, and a controller coupled with the memory array, the memory bank, and configured to interface with a host. In some examples, the controller may include a memory interface component configured to receive a request for first data located in a memory page of the memory array, a prefetch component configured to identify a first value associated with a first portion of the memory page of the memory array, and a data management component configured to communicate, to the memory bank, second data corresponding to the first portion of the memory page based at least in part on the first value of the prefetch component, the second data comprising the first data.

[0159] In some examples, the apparatus may include an access management component comprising a saturation counter that is configured to monitor a quantity of access operations performed on one or more portions of the memory page, wherein the first value is based at least in part on the quantity of access operations. In some examples, the apparatus may

include a buffer configured according to the second memory page size. In some examples, the data management component is further configured to communicate, to the buffer, third data corresponding to a second portion of the memory page.

[0160] In some examples, the apparatus may include a prefetch counter configured to store the first value. In some examples, the prefetch component comprises the prefetch counter and is further configured to modify the first value of the prefetch counter based at least in part on a first size of the first portion of the memory page and a second size of the second portion of the memory page. In some examples, the data management component is further configured to determine that the second portion of the memory page is different than the first portion of the memory page, and the prefetch component is further configured to modify the first value of the prefetch counter based at least in part on the determination. In some examples, the prefetch component is further configured to adjust the first value of the prefetch counter based at least in part on the determination.

[0161] An apparatus is described. In some examples, the apparatus may include a memory array configured according to a first memory page size, a memory bank configured according to a second memory page size different than the first memory page size, a prefetch counter configured to store a first value associated with a memory page of the memory array, and an interface controller configured to receive a first request for first data located in the memory page of the memory array and to communicate second data corresponding to a first portion of the memory page to the memory bank based at least in part on the first value of the prefetch counter, wherein the second data comprises the first data.

[0162] In some examples, the apparatus may include a buffer configured according to the second memory page size. In some examples, the interface controller may be configured to communicate, to the buffer, third data corresponding to a second portion of the memory page. In some examples, the apparatus may include a prefetch component comprising the prefetch counter and configured to modify the first value of the prefetch counter based at least in part on a relationship between the first portion of the memory page and the second portion of the memory page. In some examples, the prefetch component may be configured to determine that the second portion of the memory page is different than the first portion of the memory page and modify the first value of the prefetch counter based at least in part on the determining.

[0163] In some examples, the buffer may be configured to evict the second data. In some examples, the prefetch counter may be configured to modify the first value of the prefetch counter based at least in part on the eviction. In some examples, the apparatus may include an access management component configured to monitor a first quantity of access operations performed on the first portion of the memory page, wherein the first value of the prefetch counter based at least in part on the eviction. In some examples, the apparatus may include a saturation counter configured to monitor a second quantity of access operations performed on a subset of the first portion of the memory page, wherein the access management component comprises the saturation counter. In some examples, the interface controller may be configured to communicate, to a buffer, data corresponding to the subset of the first portion of the memory page based at least in part on the monitoring.

[0164] An apparatus is described. In some examples, the apparatus may include a memory subsystem. In some examples, the memory subsystem may include a memory array configured to store first data in a first memory page according to a first memory protocol, a prefetch counter configured to store a first value that corresponds to a first portion of the first memory page, and an interface controller configured to access the first portion of the first memory page based at least in part on the first value of the prefetch counter, wherein the memory subsystem is configured to be accessed by a host according to a second memory protocol different than the first memory protocol.

[0165] In some examples, the first memory protocol is associated with a first memory page size and first timing parameters and wherein the second memory protocol is associated with a second memory page size different than the first memory page size and second timing parameters different than the first timing parameters, the host is external to the memory subsystem and configured to access the memory subsystem according to the second memory protocol, and the interface controller is configured to communicate the first data to the host. In some examples, the apparatus may include a bus coupled with the host and the memory subsystem, the bus configured to convey signaling for the second memory protocol.

[0166] In some examples, the memory subsystem may include a memory bank configured to store, in a second memory page according to the second memory protocol, second data corresponding to the first portion of the first memory page, wherein the interface controller is configured to communicate the second to the memory bank. In some examples, the memory subsystem may include a saturation counter configured to monitor a quantity of

access operations performed on a subset of the second memory page of the memory bank, wherein the interface controller is configured to access, in the memory bank, third data corresponding to a second portion of the first memory page based at least in part on a value of the saturation counter and a buffer configured to store the third data in a third memory page according to the second memory protocol, wherein the interface controller is configured to communicate the third data to the buffer. In some examples, the memory subsystem may include a prefetch component configured to modify the first value of the prefetch counter based at least in part on a relationship between the first portion of the first memory page and the second portion of the first memory page.

[0167] It should be noted that the methods described above describe possible implementations, and that the operations and the steps may be rearranged or otherwise modified and that other implementations are possible. Furthermore, features from two or more of the methods may be combined.

[0168] Information and signals described herein may be represented using any of a variety of different technologies and techniques. For example, data, instructions, commands, information, signals, bits, symbols, and chips that may be referenced throughout the above description may be represented by voltages, currents, electromagnetic waves, magnetic fields or particles, optical fields or particles, or any combination thereof. Some drawings may illustrate signals as a single signal; however, it will be understood by a person of ordinary skill in the art that the signal may represent a bus of signals, where the bus may have a variety of bit widths.

[0169] As used herein, the term "virtual ground" refers to a node of an electrical circuit that is held at a voltage of approximately zero volts (0V) but that is not directly connected with ground. Accordingly, the voltage of a virtual ground may temporarily fluctuate and return to approximately 0V at steady state. A virtual ground may be implemented using various electronic circuit elements, such as a voltage divider consisting of operational amplifiers and resistors. Other implementations are also possible. "Virtual grounding" or "virtually grounded" means connected to approximately 0V.

[0170] The term "electronic communication" and "coupled" refer to a relationship between components that support electron flow between the components. This may include a direct connection between components or may include intermediate components. Components in electronic communication or coupled to one another may be actively

exchanging electrons or signals (e.g., in an energized circuit) or may not be actively exchanging electrons or signals (e.g., in a de-energized circuit) but may be configured and operable to exchange electrons or signals upon a circuit being energized. By way of example, two components physically connected via a switch (e.g., a transistor) are in electronic communication or may be coupled regardless of the state of the switch (i.e., open or closed).

[0171] A transistor or transistors discussed herein may represent a field-effect transistor (FET) and comprise a three terminal device including a source, drain, and gate. The terminals may be connected to other electronic elements through conductive materials, e.g., metals. The source and drain may be conductive and may comprise a heavily-doped, e.g., degenerate, semiconductor region. The source and drain may be separated by a lightly-doped semiconductor region or channel. If the channel is n-type (i.e., majority carriers are electrons), then the FET may be referred to as a n-type FET. If the channel is p-type (i.e., majority carriers are holes), then the FET may be referred to as a p-type FET. The channel may be capped by an insulating gate oxide. The channel conductivity may be controlled by applying a voltage to the gate. For example, applying a positive voltage or negative voltage to an n-type FET or a p-type FET, respectively, may result in the channel becoming conductive. A transistor may be "on" or "activated" when a voltage greater than or equal to the transistor's threshold voltage is applied to the transistor gate. The transistor may be "off" or "deactivated" when a voltage less than the transistor's threshold voltage is applied to the transistor gate.

[0172] The description set forth herein, in connection with the appended drawings, describes example configurations and does not represent all the examples that may be implemented or that are within the scope of the claims. The term "exemplary" used herein means "serving as an example, instance, or illustration," and not "preferred" or "advantageous over other examples." The detailed description includes specific details for the purpose of providing an understanding of the described techniques. These techniques, however, may be practiced without these specific details. In some instances, well-known structures and devices are shown in block diagram form to avoid obscuring the concepts of the described examples.

[0173] In the appended figures, similar components or features may have the same reference label. Further, various components of the same type may be distinguished by following the reference label by a dash and a second label that distinguishes among the

similar components. If just the first reference label is used in the specification, the description is applicable to any one of the similar components having the same first reference label irrespective of the second reference label.

[0174] Information and signals described herein may be represented using any of a variety of different technologies and techniques. For example, data, instructions, commands, information, signals, bits, symbols, and chips that may be referenced throughout the above description may be represented by voltages, currents, electromagnetic waves, magnetic fields or particles, optical fields or particles, or any combination thereof.

[0175] The various illustrative blocks and modules described in connection with the disclosure herein may be implemented or performed with a general-purpose processor, a DSP, an ASIC, an FPGA or other programmable logic device, discrete gate or transistor logic, discrete hardware components, or any combination thereof designed to perform the functions described herein. The illustrative blocks and modules, including memory controllers, may be represented as instructions in IP blocks for use in, for example, ASIC design. A general-purpose processor may be a microprocessor, but in the alternative, the processor may be any conventional processor, controller, microcontroller, or state machine. A processor may also be implemented as a combination of computing devices (e.g., a combination of a digital signal processor (DSP) and a microprocessor, multiple microprocessors, one or more microprocessors in conjunction with a DSP core, or any other such configuration).

[0176] The functions described herein may be implemented in hardware, software executed by a processor, firmware, or any combination thereof. If implemented in software executed by a processor, the functions may be stored on or transmitted over as one or more instructions or code on a computer-readable medium. Other examples and implementations are within the scope of the disclosure and appended claims. For example, due to the nature of software, functions described above can be implemented using software executed by a processor, hardware, firmware, hardwiring, or combinations of any of these. Features implementing functions may also be physically located at various positions, including being distributed such that portions of functions are implemented at different physical locations. Also, as used herein, including in the claims, "or" as used in a list of items (for example, a list of items prefaced by a phrase such as "at least one of" or "one or more of") indicates an inclusive list such that, for example, a list of at least one of A, B, or C means A or B or C or

AB or AC or BC or ABC (i.e., A and B and C). Also, as used herein, the phrase "based on" shall not be construed as a reference to a closed set of conditions. For example, an exemplary step that is described as "based on condition A" may be based on both a condition A and a condition B without departing from the scope of the present disclosure. In other words, as used herein, the phrase "based on" shall be construed in the same manner as the phrase "based at least in part on."

[0177] Computer-readable media includes both non-transitory computer storage media and communication media including any medium that facilitates transfer of a computer program from one place to another. A non-transitory storage medium may be any available medium that can be accessed by a general purpose or special purpose computer. By way of example, and not limitation, non-transitory computer-readable media can comprise RAM, ROM, electrically erasable programmable read only memory (EEPROM), compact disk (CD) ROM or other optical disk storage, magnetic disk storage or other magnetic storage devices, or any other non-transitory medium that can be used to carry or store desired program code means in the form of instructions or data structures and that can be accessed by a general-purpose or special-purpose computer, or a general-purpose or special-purpose processor. Also, any connection is properly termed a computer-readable medium. For example, if the software is transmitted from a website, server, or other remote source using a coaxial cable, fiber optic cable, twisted pair, digital subscriber line (DSL), or wireless technologies such as infrared, radio, and microwave, then the coaxial cable, fiber optic cable, twisted pair, digital subscriber line (DSL), or wireless technologies such as infrared, radio, and microwave are included in the definition of medium. Disk and disc, as used herein, include CD, laser disc, optical disc, digital versatile disc (DVD), floppy disk and Blu-ray disc where disks usually reproduce data magnetically, while discs reproduce data optically with lasers. Combinations of the above are also included within the scope of computer-readable media.

[0178] The description herein is provided to enable a person skilled in the art to make or use the disclosure. Various modifications to the disclosure will be readily apparent to those skilled in the art, and the generic principles defined herein may be applied to other variations without departing from the scope of the disclosure. Thus, the disclosure is not limited to the examples and designs described herein, but is to be accorded the broadest scope consistent with the principles and novel features disclosed herein.

CLAIMS

What is claimed is:

1. A method, comprising:

receiving a first request for first data located in a memory page of a memory array of a device;

identifying a first value of a prefetch counter associated with the memory page of the memory array; and

communicating, from the memory array to a memory bank of the device, a first portion of the memory page based at least in part on the first value of the prefetch counter, the first portion of the memory page comprising the first data.

2. The method of claim 1, further comprising:

communicating, to a buffer of the device, a second portion of the memory page, the second portion comprising at least a subset of the first data; and

modifying the first value of the prefetch counter based at least in part on a relationship between the first portion of the memory page and the second portion of the memory page.

3. The method of claim 2, further comprising:

determining that the second portion of the memory page is different from the first portion of the memory page, wherein the first value of the prefetch counter is modified based at least in part on determining that the second portion of the memory page is different from the first portion of the memory page.

4. The method of claim 3, further comprising:

evicting, by the buffer, the second portion of the memory page, wherein the first value of the prefetch counter is modified based at least in part on evicting the second portion of the memory page.

5. The method of claim 3, wherein modifying the first value of the prefetch counter comprises:

adjusting the first value of the prefetch counter based at least in part on determining that the first portion of the memory page is different than the second portion of the memory page.

6. The method of claim 1, further comprising:

writing the first portion of the memory page to the memory bank; and

monitoring a number of access operations performed on the first portion of the memory page, wherein a second value of the prefetch counter is determined based at least in part on the number of access operations.

7. The method of claim 6, wherein monitoring the number of access operations performed comprises:

monitoring the number of access operations performed on one or more subsets of the first portion of the memory page.

8. The method of claim 7, further comprising:

determining that a first number of access operations performed on a subset of the one or more subsets satisfies a threshold; and

communicating at least the subset to a buffer of the device based at least in part on the determining.

9. The method of claim 1, further comprising:

receiving a second request for second data located in the memory page of the memory array; and

communicating, from the memory array to the memory bank, a second portion of the memory page, wherein the first portion of the memory page is different from the second portion of the memory page, and wherein the second portion of the memory page comprises the second data.

10. The method of claim 9, further comprising:

communicating, from the memory bank to a buffer of the device, a third portion of the memory page; and

modifying the first value of the prefetch counter based at least in part on a relationship between the first portion of the memory page, the second portion of the memory page, or the third portion of the memory page, or any combination thereof.

11. The method of claim 1, further comprising:

receiving a second request for second data located in the memory page of the memory array;

identifying a second value of the prefetch counter associated with the memory page, wherein the second value is different from the first value; and

communicating, from the memory array to the memory bank, a second portion of the memory page based at least in part on the second value of the prefetch counter, wherein the first portion of the memory page is different from the second portion of the memory page, and wherein the second portion of the memory page comprises the second data.

12. An apparatus, comprising:

a memory array configured to store data in a first memory page configured with a first memory page size;

a memory bank configured to store data in a second memory page configured with a second memory page size different from the first memory page size; and

a prefetch counter associated with the first memory page, the prefetch counter configured to store a first value that corresponds to a first portion of the first memory page provided to the memory bank in response to a request for data located in the first memory page.

13. The apparatus of claim 12, further comprising:

a buffer configured to receive a second portion of the first memory page, wherein the first value of the prefetch counter is based at least in part on the second portion of the first memory page.

14. The apparatus of claim 13, further comprising:

a controller configured to modify the first value of the prefetch counter based at least in part on a relationship between the first portion of the first memory page and the second portion of the first memory page.

15. The apparatus of claim 13, further comprising:

a second counter configured to monitor a number of access operations performed on the first portion of the first memory page, wherein the second portion of the first memory page is based at least in part on a second value of the second counter.

16. The apparatus of claim 15, further comprising:

a controller configured to monitor access operations performed on the first portion of the first memory page.

17. The apparatus of claim 12, wherein the first memory page is partitioned into a plurality of subsets of the first memory page.

18. The apparatus of claim 12, wherein the memory array is configured to store a plurality of memory pages each configured with the first memory page size, the apparatus further comprising:

a plurality of prefetch counters each associated with a corresponding memory page of the plurality of memory pages, wherein each of the plurality of prefetch counters is configured to store a respective value corresponding to a respective portion of a respective memory page, and wherein the respective portion is responsive to a respective request for data located in the respective memory page.

19. An apparatus, comprising:

a memory array configured to store data in a first memory page configured with a first memory page size;

a memory bank configured to store data in a second memory page configured with a second memory page size different from the first memory page size;

a prefetch counter, wherein a first value of the prefetch counter is associated with the first memory page; and

a controller coupled with the memory array, the memory bank, and configured to interface with a host, the controller operable to:

receive a first request for first data located in the first memory page;

identify the first value of the prefetch counter; and

communicate, from the memory array to the memory bank, a first portion of the first memory page based at least in part on the first value of the prefetch counter, the first portion of the first memory page comprising the first data.

20. The apparatus of claim 19, further comprising:

a buffer, wherein the controller is further operable to:

communicate, from the memory bank to the buffer, a second portion of the first memory page.

21. The apparatus of claim 20, wherein the controller is operable to:

modify the first value of the prefetch counter based at least in part on a relationship between the first portion of the first memory page and the second portion of the first memory page.

22. The apparatus of claim 20, wherein the controller is operable to:

determine that the second portion of the first memory page is different from the first portion of the first memory page, wherein modifying the first value of the prefetch counter is based at least in part on determining that the second portion of the first memory page is different from the first portion of the first memory page.

23. The apparatus of claim 22, wherein the controller is operable to:

adjust the first value of the prefetch counter based at least in part on determining that the first portion of the first memory page is different than the second portion of the first memory page.

24. The apparatus of claim 19, wherein the controller is operable to:

monitor a number of access operations performed on the first portion of the first memory page, wherein a second value of the prefetch counter is determined based at least in part on the number of access operations.

25. The apparatus of claim 24, wherein the controller is operable to:

monitor the number of access operations performed on one or more subsets of the first portion of the first memory page.

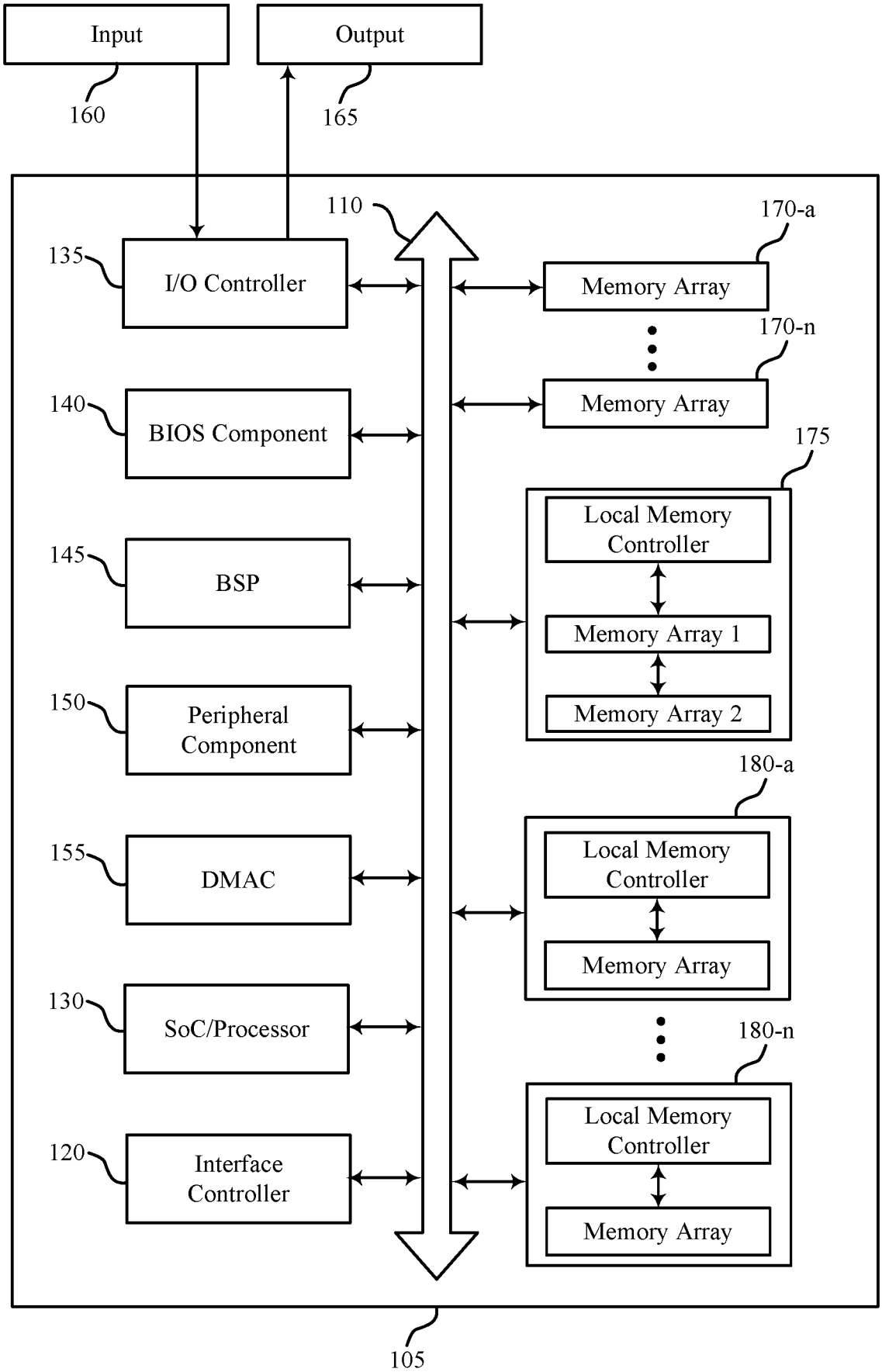
Input
Output
160
165
110
135
I/O Controller
140
BIOS Component
145
BSP
150
Peripheral Component
155
DMAC
130
SoC/Processor
120
Interface Controller
170-a
Memory Array
170-n
Memory Array
175
Local Memory Controller
Memory Array 1
Memory Array 2
180-a
Local Memory Controller
Memory Array
180-n
Local Memory Controller
Memory Array
105
100

FIG. 1

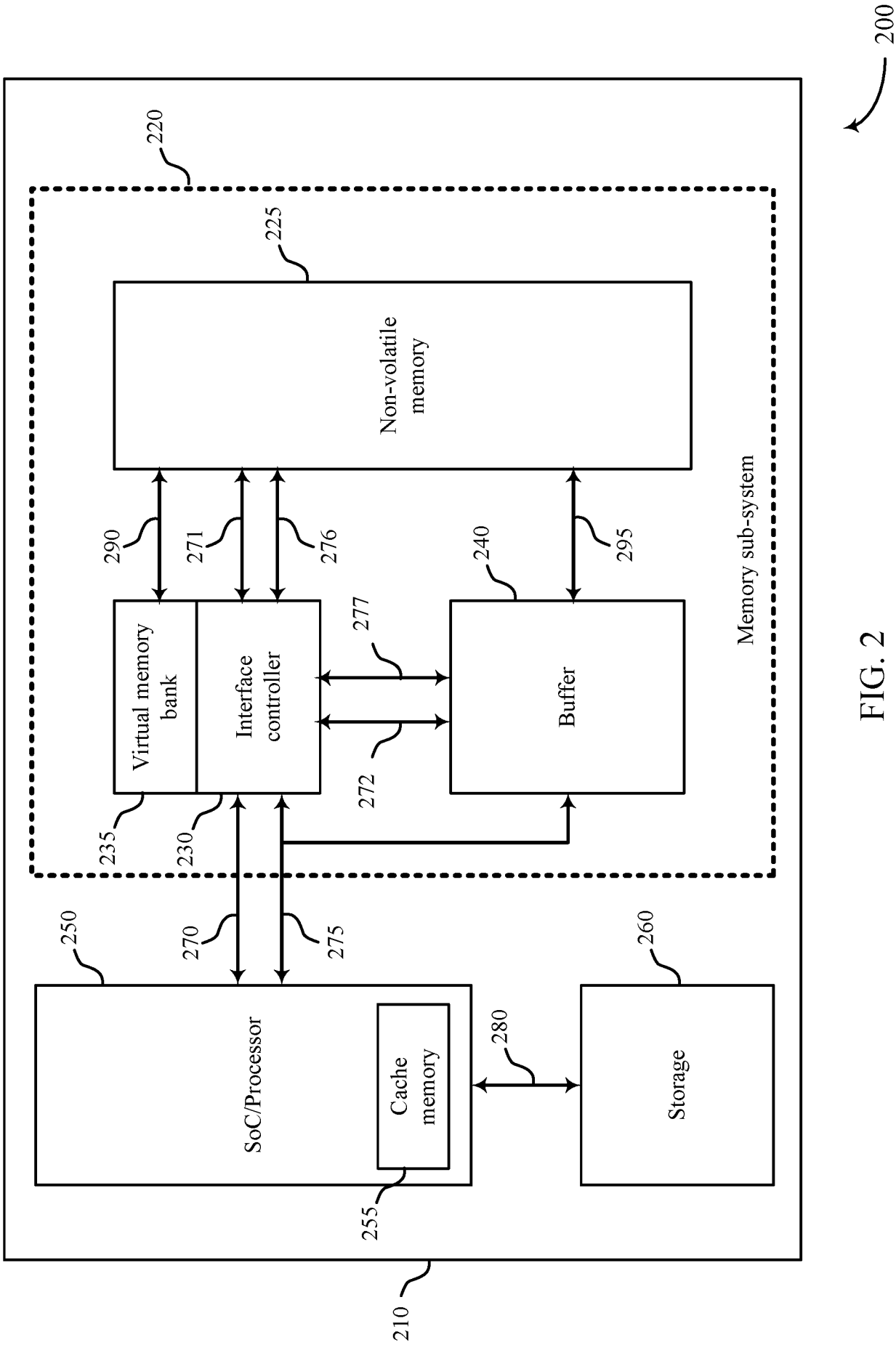
200
210
220
225
Non-volatile memory
290
271
276
240
295
Memory sub-system
Virtual memory bank
Interface controller
277
Buffer
272
235
230
250
270
275
260
SoC/Processor
Cache memory
280
Storage
255

FIG. 2

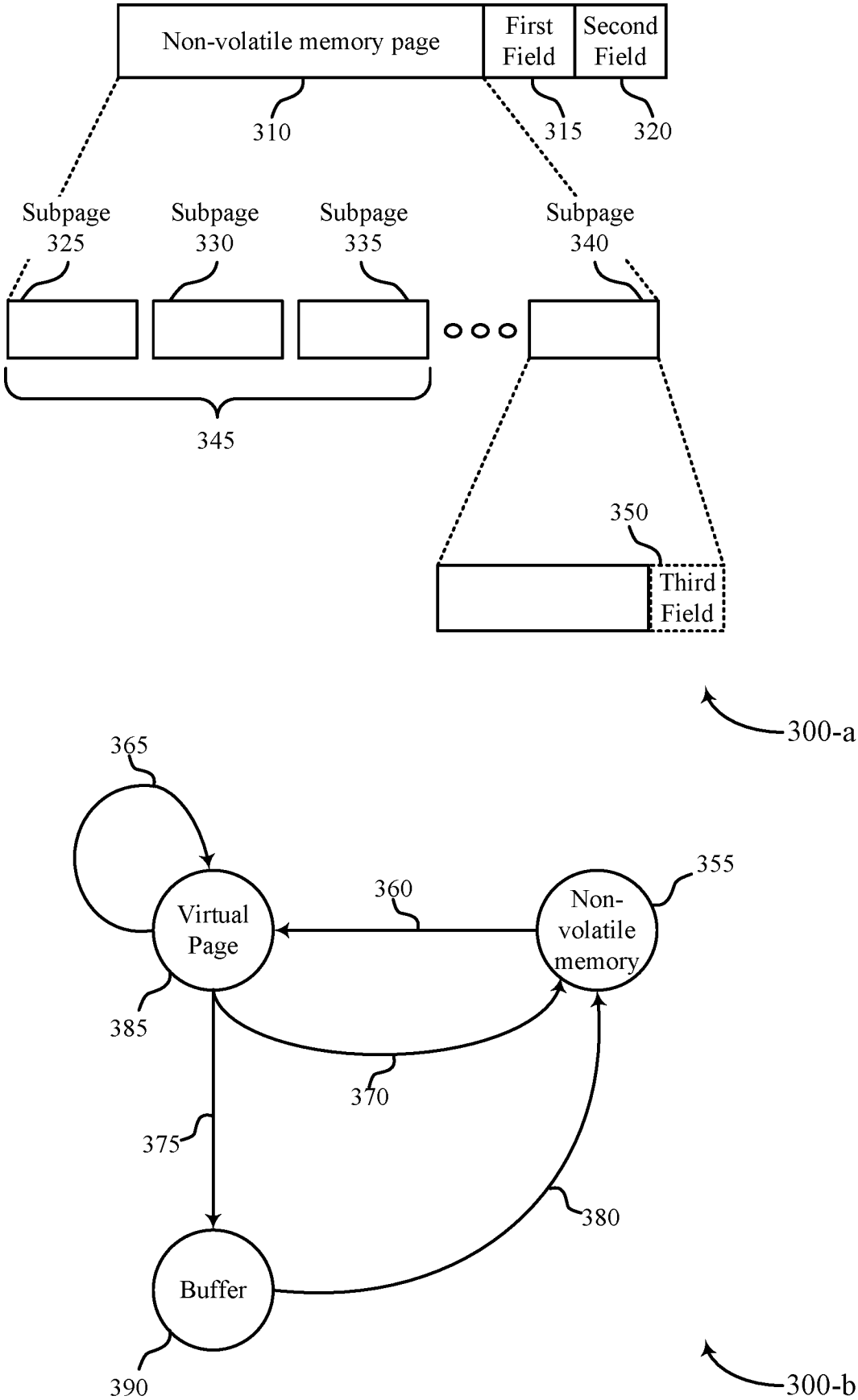
Non-volatile memory page
First Field
Second Field
310
315
320
Subpage 325
Subpage 330
Subpage 335
Subpage 340
345
350
Third Field
300-a
365
Virtual Page
360
Non-volatile memory
355
385
370
375
380
Buffer
390
300-b

FIG. 3

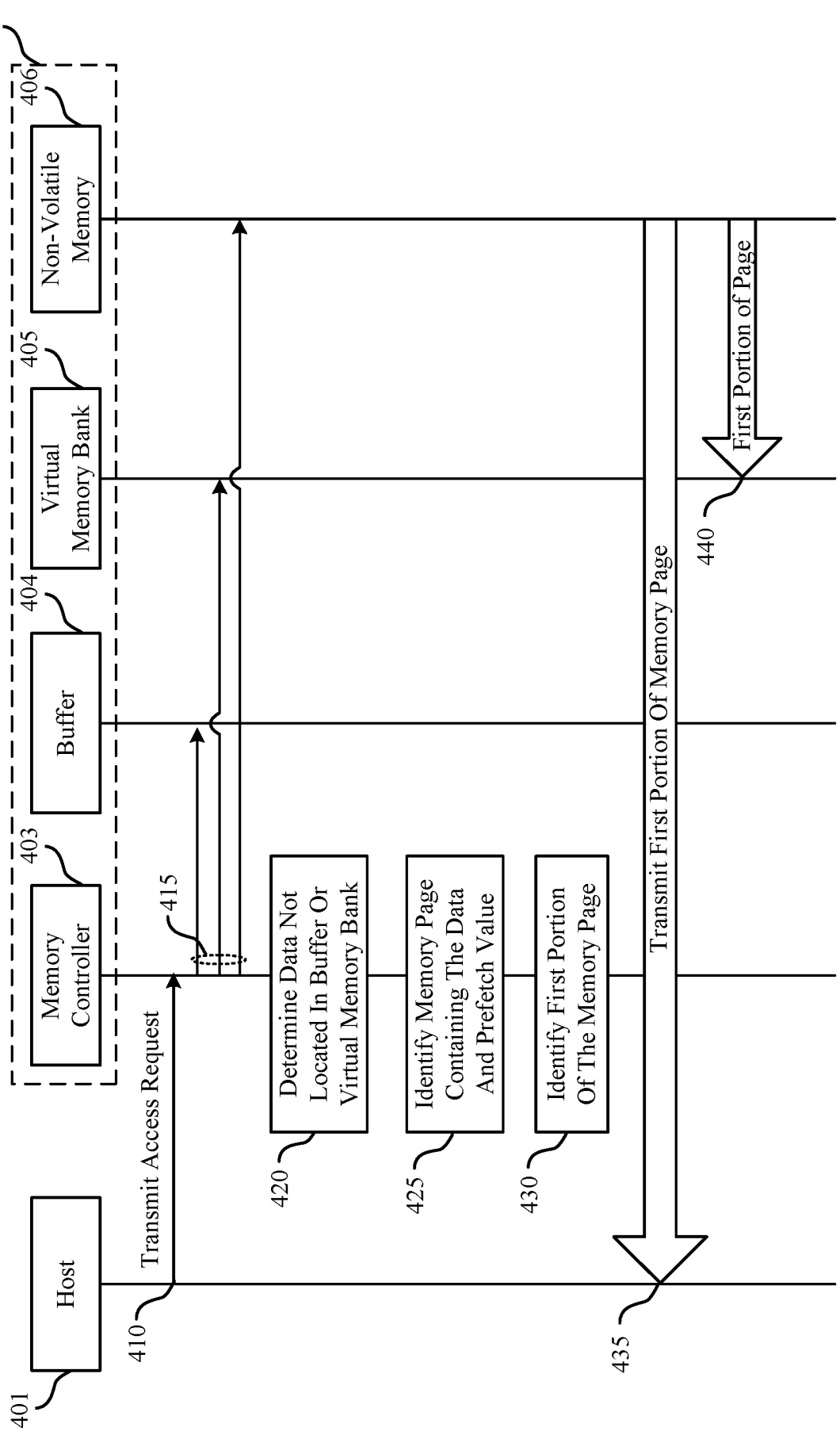
400-a
402
406
Non-Volatile Memory
405
Virtual Memory Bank
404
Buffer
403
Memory Controller
415
Host
401
Transmit Access Request
410
Determine Data Not Located In Buffer Or Virtual Memory Bank
420
Identify Memory Page Containing The Data And Prefetch Value
425
Identify First Portion Of The Memory Page
430
Transmit First Portion Of Memory Page
435
First Portion of Page
440

FIG. 4A

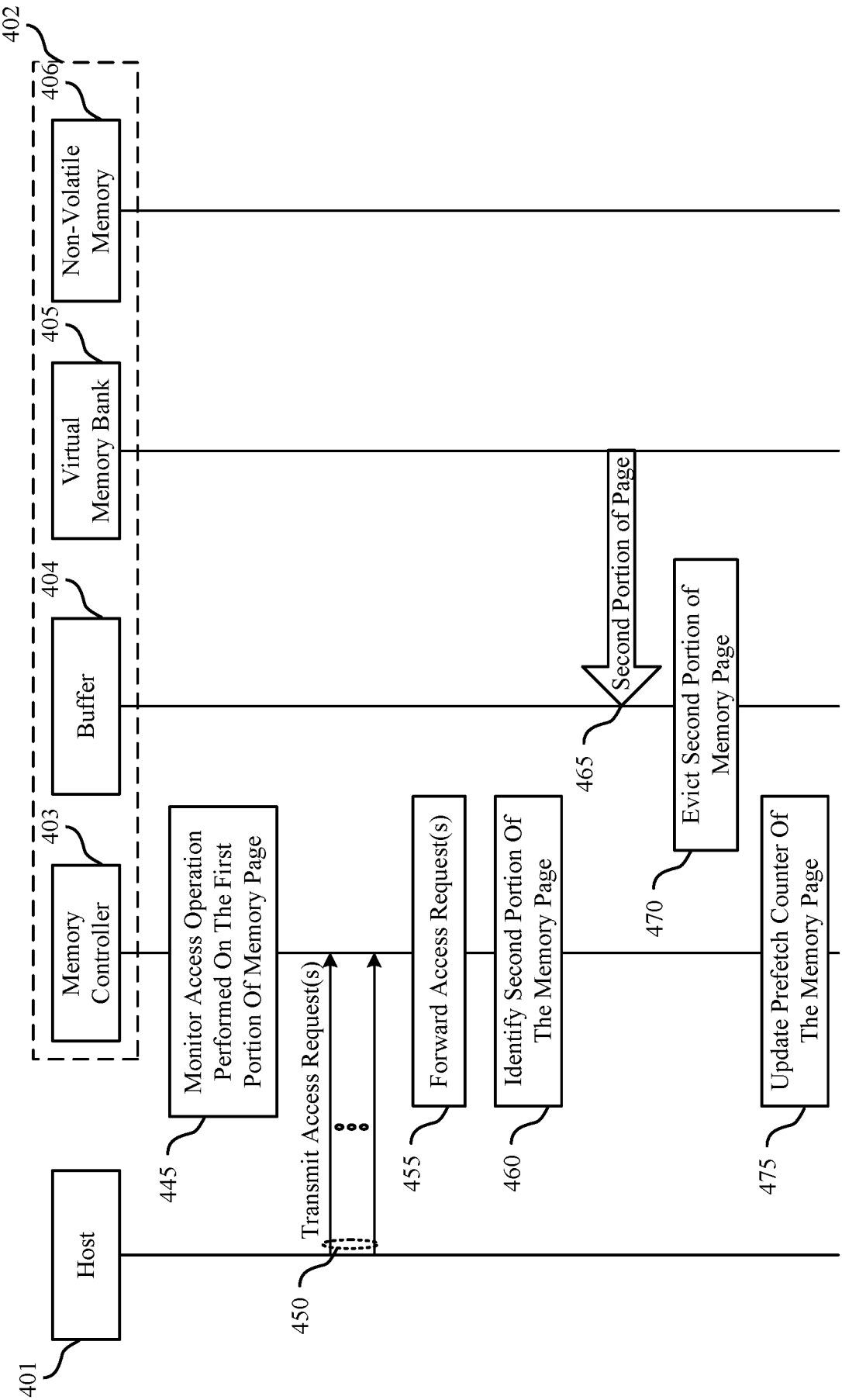
400-b
402
406
Non-Volatile Memory
405
Virtual Memory Bank
404
Buffer
403
Memory Controller
Monitor Access Operation Performed On The First Portion Of Memory Page
445
Transmit Access Request(s)
450
Forward Access Request(s)
455
Identify Second Portion Of The Memory Page
460
Second Portion of Page
465
Evict Second Portion of Memory Page
470
Update Prefetch Counter Of The Memory Page
475
Host
401

FIG. 4B

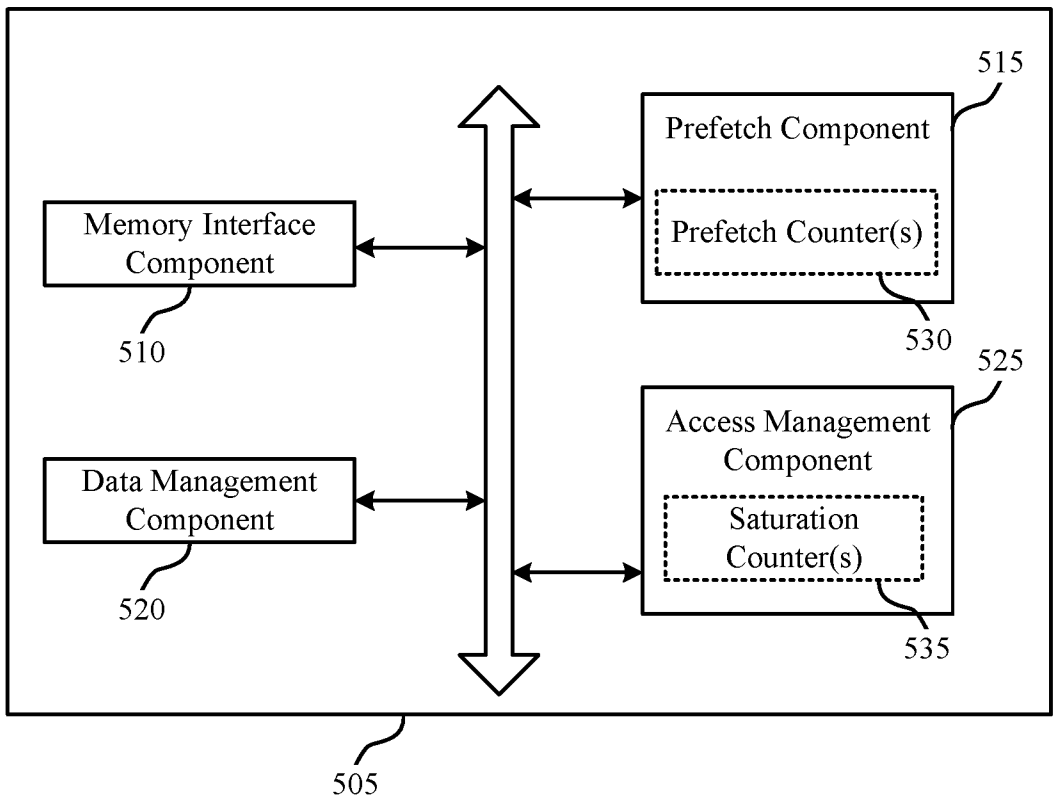
515
Prefetch Component
Prefetch Counter(s)
530
Memory Interface Component
510
525
Access Management Component
Saturation Counter(s)
535
Data Management Component
520
505
500

FIG. 5

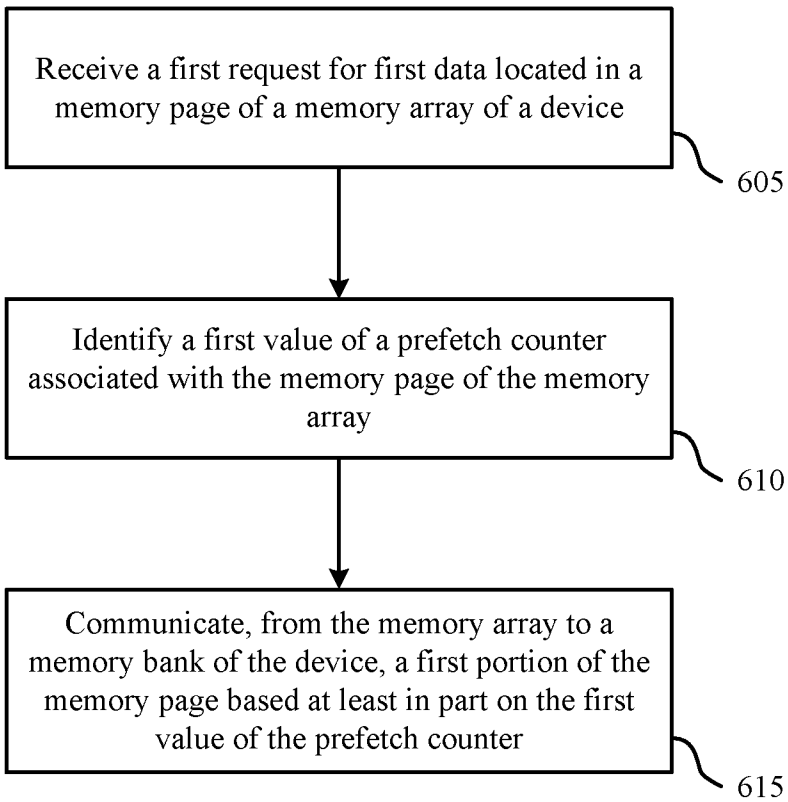
Receive a first request for first data located in a memory page of a memory array of a device
605
Identify a first value of a prefetch counter associated with the memory page of the memory array
610
Communicate, from the memory array to a memory bank of the device, a first portion of the memory page based at least in part on the first value of the prefetch counter
615
600

FIG. 6

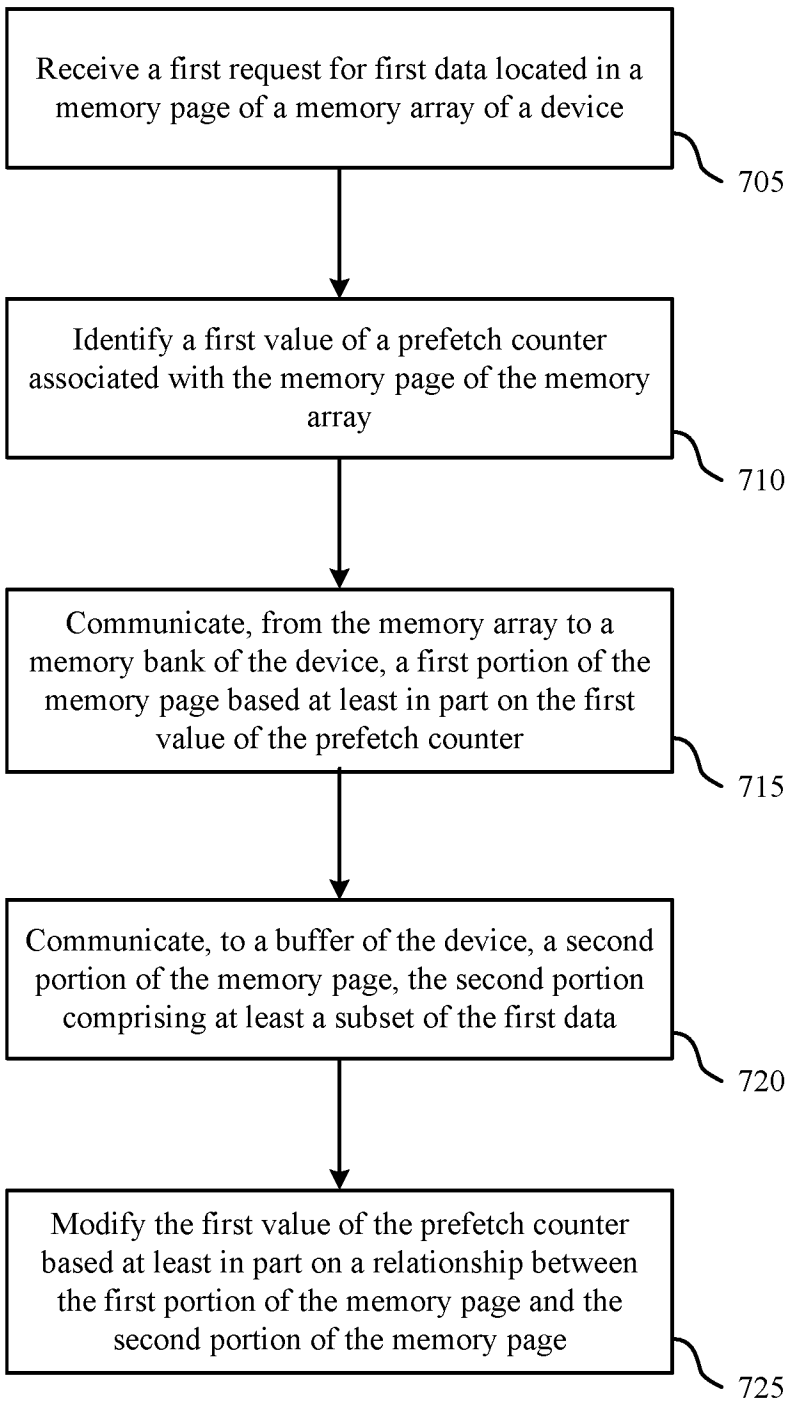
Receive a first request for first data located in a memory page of a memory array of a device
705
Identify a first value of a prefetch counter associated with the memory page of the memory array
710
Communicate, from the memory array to a memory bank of the device, a first portion of the memory page based at least in part on the first value of the prefetch counter
715
Communicate, to a buffer of the device, a second portion of the memory page, the second portion comprising at least a subset of the first data
720
Modify the first value of the prefetch counter based at least in part on a relationship between the first portion of the memory page and the second portion of the memory page
725
700

FIG. 7

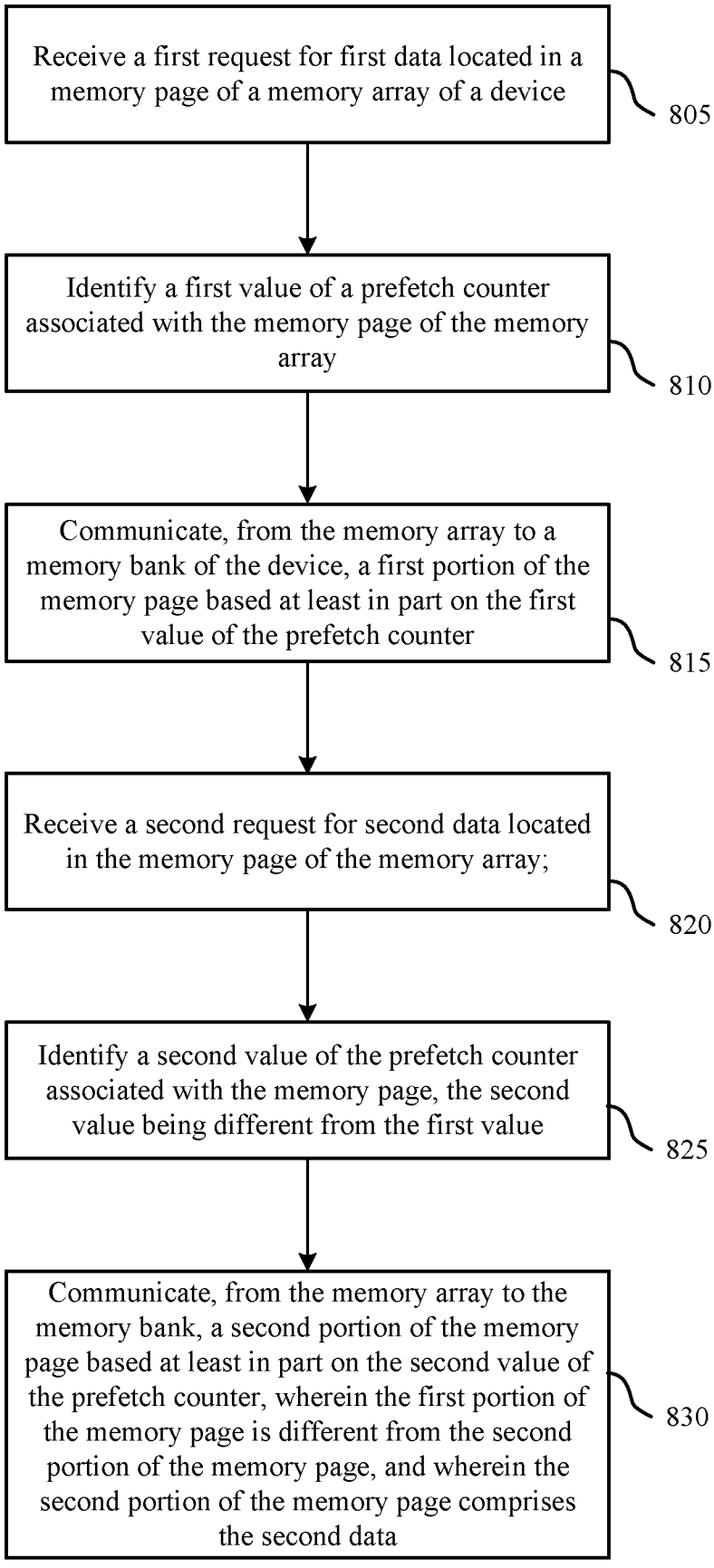
Receive a first request for first data located in a memory page of a memory array of a device
805
Identify a first value of a prefetch counter associated with the memory page of the memory array
810
Communicate, from the memory array to a memory bank of the device, a first portion of the memory page based at least in part on the first value of the prefetch counter
815
Receive a second request for second data located in the memory page of the memory array;
820
Identify a second value of the prefetch counter associated with the memory page, the second value being different from the first value
825
Communicate, from the memory array to the memory bank, a second portion of the memory page based at least in part on the second value of the prefetch counter, wherein the first portion of the memory page is different from the second portion of the memory page, and wherein the second portion of the memory page comprises the second data
830
800

FIG. 8

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2019/028688

A. CLASSIFICATION OF SUBJECT MATTER

G06F 12/0862(2016.01)i, G06F 12/02(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 12/0862; G06F 12/00; G06F 12/02; G06F 12/1081; G06F 3/06; G11C 11/34

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: prefetch, memory, page, array, bank, counter, value, portion, buffer, modify, adjust, evict, monitor, size

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2006-0053256 A1 (WILLIAM C. MOYER et al.) 09 March 2006 See paragraphs [0014]-[0028]; claims 1, 9, 15, 17; and figures 1-4.	1-25
Y	US 2004-0141374 A1 (JOO WEON PARK et al.) 22 July 2004 See paragraphs [0024], [0067]; claims 1, 15; and figure 3.	1-25
A	US 2018-0121106 A1 (SAMSUNG ELECTRONICS CO., LTD.) 03 May 2018 See paragraphs [0045]-[0061]; and figure 4.	1-25
A	US 8959307 B1 (REY H. BRUCE et al.) 17 February 2015 See column 7, line 3 - column 8, line 55; claim 1; and figure 3.	1-25
A	US 2014-0115294 A1 (INTERNATIONAL BUSINESS MACHINES CORPORATION) 24 April 2014 See paragraphs [0021]-[0025]; claims 1-3, 5; and figure 1.	1-25

☐ 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
29 August 2019 (29.08.2019)	**29 August 2019 (29.08.2019)**

Name and mailing address of the ISA/KR
International Application Division
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon, 35208, Republic of Korea
Facsimile No. +82-42-481-8578

Authorized officer
CHIN, Sang Bum
Telephone No. +82-42-481-8398

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2019/028688

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2006-0053256 A1	09/03/2006	CN 100407165 C	30/07/2008
		CN 1813246 A	02/08/2006
		EP 1652092 A2	03/05/2006
		JP 2007-500897 A	18/01/2007
		JP 4875981 B2	15/02/2012
		KR 10-1093317 B1	14/12/2011
		KR 10-2006-0052936 A	19/05/2006
		TW 200519604 A	16/06/2005
		TW I352293 B	11/11/2011
		US 7200719 B2	03/04/2007
		WO 2005-013039 A2	10/02/2005
		WO 2005-013039 A3	21/07/2005
US 2004-0141374 A1	22/07/2004	US 6775184 B1	10/08/2004
US 2018-0121106 A1	03/05/2018	CN 108021334 A	11/05/2018
		KR 10-2018-0049338 A	11/05/2018
US 8959307 B1	17/02/2015	US 10120586 B1	06/11/2018
US 2014-0115294 A1	24/04/2014	CN 103777898 A	07/05/2014
		CN 103777898 B	14/09/2016
		US 9251048 B2	02/02/2016