

(19) World Intellectual Property
Organization
International Bureau



(43) International Publication Date
22 July 2004 (22.07.2004)

PCT

(10) International Publication Number
WO 2004/061672 A2

(51) International Patent Classification⁷: **G06F 12/00**

(21) International Application Number:
PCT/US2003/039155

(22) International Filing Date:
10 December 2003 (10.12.2003)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
10/355,485 31 December 2002 (31.12.2002) US

(71) Applicant: **INTEL CORPORATION** [US/US]; 2200
Mission College Boulevard, Santa Clara, CA 95052 (US).

(72) Inventor: **CHRISTENSON, Bruce**; 2951 Buxton Court,
Forest Grove, OR 97116 (US).

(74) Agents: **WISE, Roger, R.** et al.; Pillsbury Winthrop, 725
South Figueroa Street, Suite 2800, Los Angeles, CA 90017
(US).

(81) Designated States (*national*): AE, AG, AL, AM, AT, AU,
AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU,
CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH,
GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC,
LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW,
MX, MZ, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC,
SD, SE, SG, SK, SL, SY, TJ, TM, TN, TR, TT, TZ, UA,
UG, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States (*regional*): ARIPO patent (BW, GH,
GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW),
Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM),
European patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE,
ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PT, RO, SE,
SI, SK, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA,
GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished
upon receipt of that report*

*For two-letter codes and other abbreviations, refer to the "Guid-
ance Notes on Codes and Abbreviations" appearing at the begin-
ning of each regular issue of the PCT Gazette.*

(54) Title: READ-WRITE SWITCHING METHOD FOR A MEMORY CONTROLLER

(57) Abstract: A system includes a memory device. The memory device has a first bank and a second bank. A memory controller has a write request queue to store write requests. When a read bank conflict exists between a first read request to the first bank and a second read request to the first bank, a first write request is executed to the second bank during a delay. The delay takes place after the first read request is executed and before the second read request is executed.



TITLE

READ-WRITE SWITCHING METHOD FOR A MEMORY CONTROLLER

BACKGROUND5 1. Technical Field

Embodiments of this invention relate to the field of memory controllers to read and write to a memory.

2. Description of the Related Arts

10 Current Double Data Rate (“DDR”) and Double Date Rate-II (“DDR-II”) Dynamic Random Access Memory (DRAM) subsystems exhibit significant latency when performing back-to-back reads to the same bank in memory. Several server chipsets can operate within a single rank of memory, providing a total of 4 independent memory banks. The probability of random traffic hitting the same bank repeatedly in such configuration is
15 1:4 (25%). In the case of page empty accesses (which are typical in server chipsets), the total bank conflict time is the total of the “activate to read” (tRCD) latency plus the “read to pre-charge” (tRPD) latency plus the “pre-charge” to “activate” (tRP) latency, or the “activate to activate command period” (tRC), whichever timing constraint is longer. In the case of DDR-266, Intel DDR-266 JEDEC Specification Addendum, Rev. 09, published
20 August 14, 2001, this bank conflict duration is typically 65 ns. In the case of DDR-II-533, published June 25, 2001, this bank conflict duration is typically 60ns. In contrast, the total data latency for a burst of 4 (32 bytes) for each memory technology is only 15 ns and 7.5 ns, respectively.

Traditional read-write switching methods prioritize read requests until the write
25 requests queue up to a preset threshold. Then, an amount of write requests are “drained”

back-to-back, and priority returns to the read requests (which are queued while waiting for the writes to drain). However, such read-write switching methods are slow and inefficient because no additional reads or writes are executed during the bank conflict duration.

5 BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 illustrates a memory system according to an embodiment of the invention;

FIG. 2 illustrates a memory controller according to an embodiment of the invention;

FIG. 3 illustrates a timing diagram according to an embodiment of the invention;

10 FIG. 4 illustrates a timing diagram of read operations having a bank conflict when a write operation is executed within the same rank according to an embodiment of the invention;

FIG. 5 illustrates a timing diagram of read operations having a bank conflict when multiple write operations are executed within the different ranks according to an

15 embodiment of the invention;

FIG. 6 illustrates a method of executing multiple write commands when there is a read bank conflict according to an embodiment of the invention;

FIG. 7A illustrates a first portion of an additional method of executing write commands according to an embodiment of the invention; and

20 FIG. 7B illustrates a second portion of an additional method of executing write commands according to an embodiment of the invention.

DETAILED DESCRIPTION

An embodiment of the invention may be utilized to more quickly service write
25 requests in DRAM memory channels. A memory device may queue write requests to

memory. When consecutive reads from the memory are to the same rank, same bank, but different pages, a “memory bank conflict” exists and a latency occurs between activate commands to the bank. While reads are scheduled, write requests may queue in a write request queue. If there is a conflict (i.e., there are consecutive reads, and the second read is to the same rank, same bank, but different page), a write operation may occur to a different bank within the same rank while the page for the read action is waiting to be activated. Depending on the time necessary for the activation of a row, additional writes may be implemented on other banks within the same rank. Such method may result in faster execution of write commands, and quicker performance of the DRAM memory device.

Alternatively, the memory device may queue read commands while writes are being executed. The read commands may be queued while the write commands are executed. Once a bank conflict occurs (i.e., two consecutive writes to the same bank in the same rank, but different pages), the read may be executed while the next page is activated. Accordingly, the DRAM may more efficiently service read and write operations.

FIG. 1 illustrates a memory system according to an embodiment of the invention. The memory system may include a memory controller 100 and a memory rank 105. The memory rank 105 may include several memory banks, such as bank A 110, bank B 115, bank C 120, and bank D 125. The memory controller 100 may be in communication with each of bank A 110, bank B 115, bank C 120, and bank D 125. Although only a single memory rank 105 is shown in FIG. 1, multiple memory ranks may also be utilized. Each bank may include an array having a plurality of pages (i.e., rows) and columns of storage locations. Data may be read from or written to a location in the array by activating a specific page. Then, data may be read from locations within the page. To read data from

a different page, the different page may be pre-charged and then activated after the data is read from the first page.

FIG. 2 illustrates a memory controller 100 according to an embodiment of the invention. The memory controller 100 may include a read request queue 200. The read request queue 200 may be utilized to queue read requests when reads are currently being made, as well as when writes are being made. The memory controller 100 may also include a write request queue 205 which may be also utilized to queue write requests when reads are currently being made, as well as when writes are being made.

The memory controller 100 may also include a write data buffer 210, which may cache data to be written in a coherent and/or consistent manner. The write data buffer 210 may store write data that has not yet been written to memory. The write data buffer 210 may be one entry deep, but is typically more than one entry deep, for performance reasons. If a read request following an earlier write request to the same memory address enters the memory controller 100, the memory controller 100 may either service the read request using the data from the write data buffer 210, or it may schedule the write request and then schedule the read request.

The memory controller 100 may include logic 220. The logic 220 may be utilized to determine whether the next pending read command has a protocol conflict with a previously scheduled command. A protocol conflict may be a condition in which a DRAM timing constraint prohibits the memory controller 100 from scheduling a DRAM command in the current clock cycle. Protocol conflicts include bank conflicts, where back to back accesses to the same rank, bank, different page are delayed a specific amount of time between active commands (due to tRC DRAM timing constraints).

The memory controller 100 may be in communication with a processor 225 to control each of the read request queue 200, the write request queue 205, the write data

buffer 210, and the logic 220. The processor 225 may, in turn, be in communication with an instruction storage device 230. The instruction storage device 230 may contain instructions (e.g., program code) that are executable by the processor 225.

The memory controller 100 may cause writes to be executed during the latency
5 caused by bank conflicts. The memory controller 100 may cause the oldest queued write to be scheduled during a bank conflict, provided it does not conflict with the current read operation.

FIG. 3 illustrates a timing diagram according to an embodiment of the invention. The scale at the top of the diagram represents time. Each of the numerical units may
10 represent 10 ns, for example. "ACT" represents an activate command. When a page is selected from which to be read, the page may be activated by the "ACT" command. As illustrated, the "ACT" command may take 3 time units for the page to become activated. This activation latency is represented by time delay "tRCD" (tRCD is an abbreviation for "time to activate for the read command"). Next, a read command (shown as "RD") may
15 be executed. The RD command may take 1 time unit to be executed, and then an additional 2 time units until the data is actually read. This latency is represented by the time delay "tCL" ("tCL" may stand for "time for CAS latency," where "CAS" is an abbreviation for "column access strobe"). The entire time from when the page is activated until data can be read is represented by the time delay "tRCD + tCL", which is the sum of
20 the tRCD and tCL timing constraints, and is shown as taking 6 time units.

Next, a precharge instruction ("PRE") is executed, and the data is read (shown as "RD DATA" on the timing diagram) from the previous RD instruction. The precharge process may take 3 time units, shown as time "tRP" (abbreviation for the precharge command duration). The precharge ("PRE") command de-activates a page in a bank.
25 This must happen before another page in the same bank has been activated with the

activate (“ACT”) command. If the page is not precharged, then the next access to that bank will either be a page hit (same rank, bank, page) or a page miss (same rank, bank, different page). Page hits require only the “RD” command. Page misses require a “PRE”, an “ACT”, and then a “RD”. If a page is precharged, the next time that bank is accessed
5 (which may be some much later time) then the next access will be a page empty access, which requires only an “ACT” and then a “RD”.

Next, if data is to be read from a different page within the same bank, an additional tRC delay is encountered. The ACT and RD commands may again be executed, and a time period of an additional 6 time units may elapse before the data from the next page
10 may be read. An additional PRE command may then be executed, etc.

Accordingly, as shown, a significant latency shown by the time period tRCD is present when consecutive reads are made from different pages within the same bank (the tRCD period would not be present if consecutive reads were made from the same page of the same bank and rank). The memory controller 100 may take advantage of this tRCD
15 latency by executing write instructions during this time period, resulting in more efficient and faster scheduling of write commands.

FIG. 4 illustrates a timing diagram of read operations having a bank conflict when a write operation is executed within the same rank according to an embodiment of the invention. As shown, the read commands and time delays are the same as those shown in
20 FIG. 3. However, in the timing diagram of FIG. 4, the memory controller 100 takes advantage of the tRCD latency by writing data to a different memory bank within the same rank. For example, if a read operation is to a page within bank A 110 of the memory rank 105, a write operation may be made to one of the other banks (e.g., bank B 115, bank C 120, or bank D 125, but not bank A 110) during the tRAS time period.

A write command ("WR") may be executed during the tRP time period during the pre-charge operation, and the data may actually be written ("WR DATA") during the next tRCD time period. tWTR is called the "write to read command delay" and the delay constraint must be satisfied if there is a read after a write to the same rank. This is why it is better for a memory controller 100 to schedule the last write to a different rank than the subsequent read. Accordingly, a WR command may be executed and data be written to another bank without adversely affecting execution of the RD command.

FIG. 5 illustrates a timing diagram of read operations having a bank conflict when multiple write operations are executed within different ranks according to an embodiment of the invention. As shown, write operations may be executed to write data to banks in different ranks. To avoid impacting the latency of the second read because of the tWTR delay, the final write transaction would need to target a different rank, than the second read. As shown, data may be read from a bank within rank "0." During the tRCD time period, data may first be written to a memory bank within rank "X", and then data may be written to a memory bank within rank "1." As shown, the WR command to write data to the memory bank with rank "X" may be executed at the end of the tRP period. Then, while the data is actually written to the memory bank, the write command WR1 may be executed, and then data may be written to the memory bank within rank "1." Accordingly, two write operations may be executed without adversely affecting the RD operation.

Other embodiments may execute two writes to the same rank interleaved between two reads with a bank conflict, and five or six writes (depending on the technology protocol timings) may be interleaved between two reads with a bank conflict, provided the last three write commands are to different ranks than the read commands. If the tRCD timing constraint remains constant, but the DRAM clock period increases (i.e., clocks faster), then the number of transactions that can be scheduled during a bank conflict

(tRCD) can be increased from 1 or 2 to many more. How many can be scheduled without impacting the latency of the second original transaction may be determined by the specific DRAM device timing parameters being used in the system.

FIG. 6 illustrates a method of executing multiple write commands when there is a read bank conflict according to an embodiment of the invention. First, the system may determine 600 whether there is a bank conflict for the next read operation. If “no,” processing may remain at operation 600. If “yes,” processing may proceed to operation 602. At operation 602, the system may load the oldest queued write command from the write request queue 205. The system may then determine 605 whether both loaded write commands are from the same rank. If “no,” processing proceeds to operation 615. If “yes,” processing proceeds to operation 610, where the newer write command is removed 610 and the next queued write command is loaded. After operation 610, processing may return to operation 605. At operation 615, the system may determine whether either loaded write command is for the same bank as the next read. If “no,” processing proceeds to operation 625. If “yes,” processing proceeds to operation 620, where the conflicting write command is removed 620, and the next queued write command is loaded. After operation 620, processing may return to operation 605. At operation 625, after the data has been read, the system may execute the older write command first, and then execute the newer write command. In other embodiments, the newer write command may be executed first. In additional embodiments, more than two write commands may be executed during a read operation bank conflict. After operation 625, processing may return to operation 600.

FIG. 7A illustrates a first portion of an additional method of executing write commands according to an embodiment of the invention. As shown, the system may first determine 700 whether the read request queue 200 and the write request queue 205 are

both empty. If “yes,” processing remains at operation 700. If “no,” processing proceeds to operation 705, where the system determines 705 whether the number of write requests in the write request queue 205 exceeds a preset threshold. The system may have a preset threshold for the number of write requests it may store. Once the number of write requests
5 exceeds the threshold level, the system may give priority to executing the write requests and may execute them as soon as possible until the number of queued write requests falls below the threshold. If the answer to operation 705 is “yes,” processing proceeds to operation 710. If the answer at operation 705 is “no,” processing proceeds to operation 730 shown in FIG. 7B.

10 At operation 710, the system may determine whether the next write request is available for execution. If “yes,” processing proceeds to operation 715. If “no,” processing proceeds to operation 720. At operation 715, the system may execute the next write request, and processing may then return to operation 700. At operation 720, the system may determine whether the next read request is available for execution. If “yes,”
15 processing proceeds to operation 725. If “no,” processing returns to operation 700. Next, at operation 725, the system executes the next write request and processing then returns to operation 700.

FIG. 7B illustrates a second portion of an additional method of executing write commands according to an embodiment of the invention. At operation 730, the system
20 may determine whether the next read request is available for execution. If “yes,” processing proceeds to operation 735. If “no,” processing proceeds to operation 740. At operation 735, the system may execute the next read request and then proceed to operation 700. Next, at operation 740, the system may determine whether the next write request is available for execution. If “yes,” processing proceeds to operation 745. If “no,”

processing proceeds to operation 700. At operation 745, the system executes the next write request and then processing returns to operation 700.

While the description above refers to particular embodiments of the present invention, it will be understood that many modifications may be made without departing
5 from the spirit thereof. The accompanying claims are intended to cover such modifications as would fall within the true scope and spirit of the present invention. The presently disclosed embodiments are therefore to be considered in all respects as illustrative and not restrictive, the scope of the invention being indicated by the appended
10 claims, rather than the foregoing description, and all changes which come within the meaning and range of equivalency of the claims are therefore intended to be embraced therein.

WHAT IS CLAIMED IS:

1. A memory system, comprising:
a memory device having a first bank and a second bank; and
a memory controller having a write request queue to store write requests, wherein
5 when a read bank conflict exists between a first read request to the first bank and a second
read request to the first bank, a first write request is executed to the second bank during a
delay, after the first read request is executed and before the second read request is
executed.
2. The memory system according to claim 1, further including a read request queue to
10 store read requests.
3. The system according to claim 1, further including a write data buffer to store write
data.
4. The system according to claim 1, further including logic to determine if one of the
second read request and the first write request has a protocol conflict with the first read
15 request.
5. The system according to claim 1, wherein the memory device is a dynamic random
access memory (DRAM).
6. A memory system, comprising:
a memory device having a first bank and a second bank; and
20 a memory controller having a read request queue to store read requests, wherein
when a write bank conflict exists between a first write request to the first bank and a
second write request to the first bank, a first read request is executed to the second bank
during a delay, after the first write request is executed and before the second write request
is executed.

7. The memory system according to claim 6, further including a write request queue to store write requests.

8. The system according to claim 6, further including logic to determine if one of the
5 second write request and the first read request has a protocol conflict with the first write request.

9. The system according to claim 6, wherein the memory device is a dynamic random access memory (DRAM).

10. A method, comprising:
10 storing write requests in a write request queue;
executing a first read request to a first bank of a memory device;
executing a second read request to the first bank, after a delay, after executing the first read request, when a read bank conflict exists between the first read request and the second read request; and
15 executing a first write request to a second bank during the delay.

11. The method of claim 10, wherein the first bank and the second bank are within a same rank.

12. The method of claim 10, wherein the first bank is not within a same rank as the second bank.

20 13. The method of claim 10, further including storing the first write request in a write request queue.

14. The method of claim 10, further including determining whether one of the second read request and the first write request has a protocol conflict with the first read request.

15. A method, comprising:
25 storing read requests in a read request queue;

executing a first write request to a first bank of a memory device;

executing a second write request to the first bank, after a delay, after executing the first write request, when a write bank conflict exists between the first write request and the second write request; and

5 executing a first read request to a second bank during the delay.

16. The method of claim 15, wherein the first bank and the second bank are within a same rank.

17. The method of claim 15, wherein the first bank is not within a same rank as the second bank.

10 18. The method of claim 15, further including storing the first read request in a read request queue.

19. The method of claim 15, further including determining whether one of the second write request and the first read request has a protocol conflict with the first write request.

20. An article comprising a storage medium having stored thereon instructions that
15 when executed by a machine result in the following:

storing write requests in a write request queue;

executing a first read request to a first bank of a memory device;

20 executing a second read request to the first bank, after a delay, after executing the first read request, when a read bank conflict exists between the first read request and the second read request; and

executing a first write request to a second bank during the delay.

21. The article of claim 20, wherein the first bank and the second bank are within a same rank.

22. The article of claim 20, wherein the first bank is not within a same rank as the
25 second bank.

23. The article of claim 20, wherein the instructions further result in storing the first write request in a write request queue.

24. The article of claim 20, wherein the instructions further result in determining whether one of the second read request and the first write request has a protocol conflict
5 with the first read request.

25. An article comprising a storage medium having stored thereon instructions that when executed by a machine result in the following:

storing read requests in a read request queue;

executing a first write request to a first bank of a memory device;

10 executing a second write request to the first bank, after a delay, after executing the first write request, when a write bank conflict exists between the first write request and the second write request; and

executing a first read request to a second bank during the delay.

26. The article of claim 25, wherein the first bank and the second bank are within a
15 same rank.

27. The article of claim 25, wherein the first bank is not within a same rank as the second bank.

28. The article of claim 25, wherein the instructions further result in storing the first read request in a read request queue.

20 29. The article of claim 25, wherein the instructions further result in determining whether one of the second write request and the first read request has a protocol conflict with the first write request.

30. A memory controller, comprising:

a write request queue to store write requests, wherein when a read bank conflict

25 exists between a first read request to a first bank of a memory device, and a second read

request to the first bank, a first write request is executed to a second bank of the memory device during a delay before the second read request is executed.

31. The memory controller according to claim 30, further including a read request queue to store read requests.

5 32. The memory controller according to claim 30, further including a write data buffer to store write data.

33. The memory controller according to claim 30, further including logic to determine if one of the second read request and the first write request has a protocol conflict with the first read request.

10 34. The system according to claim 30, wherein the memory device is a dynamic random access memory (DRAM).

35. A memory controller, comprising:

a read request queue to store read requests, wherein when a write bank conflict exists between a first write request to a first bank of a memory device, and a second write
15 request to the first bank, a first read request is executed to a second bank of the memory device, during a delay before the second write request is executed.

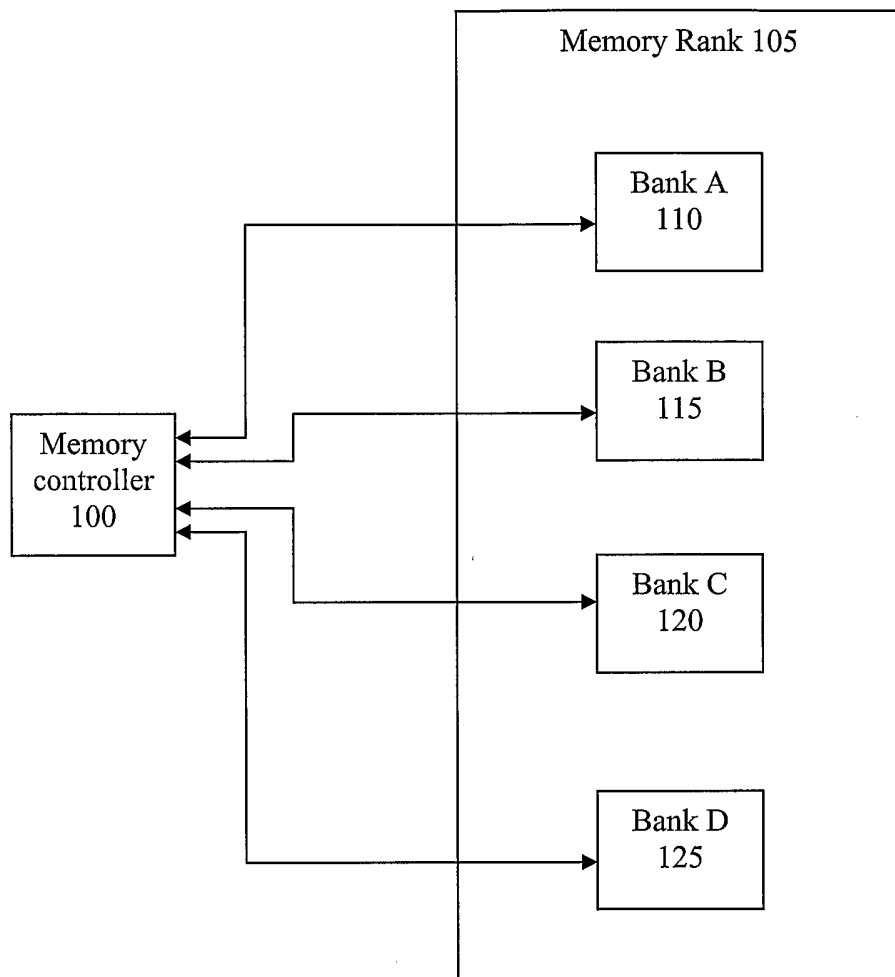
36. The memory controller according to claim 35, further including a write request queue to store read requests.

37. The memory controller according to claim 35, further including a write data buffer
20 to store write data.

38. The memory controller according to claim 35, further including logic to determine if one of the second read request and the first write request has a protocol conflict with the first read request.

39. The system according to claim 35, wherein the memory device is a dynamic
25 random access memory (DRAM).

1/7

**FIG. 1**

2/7

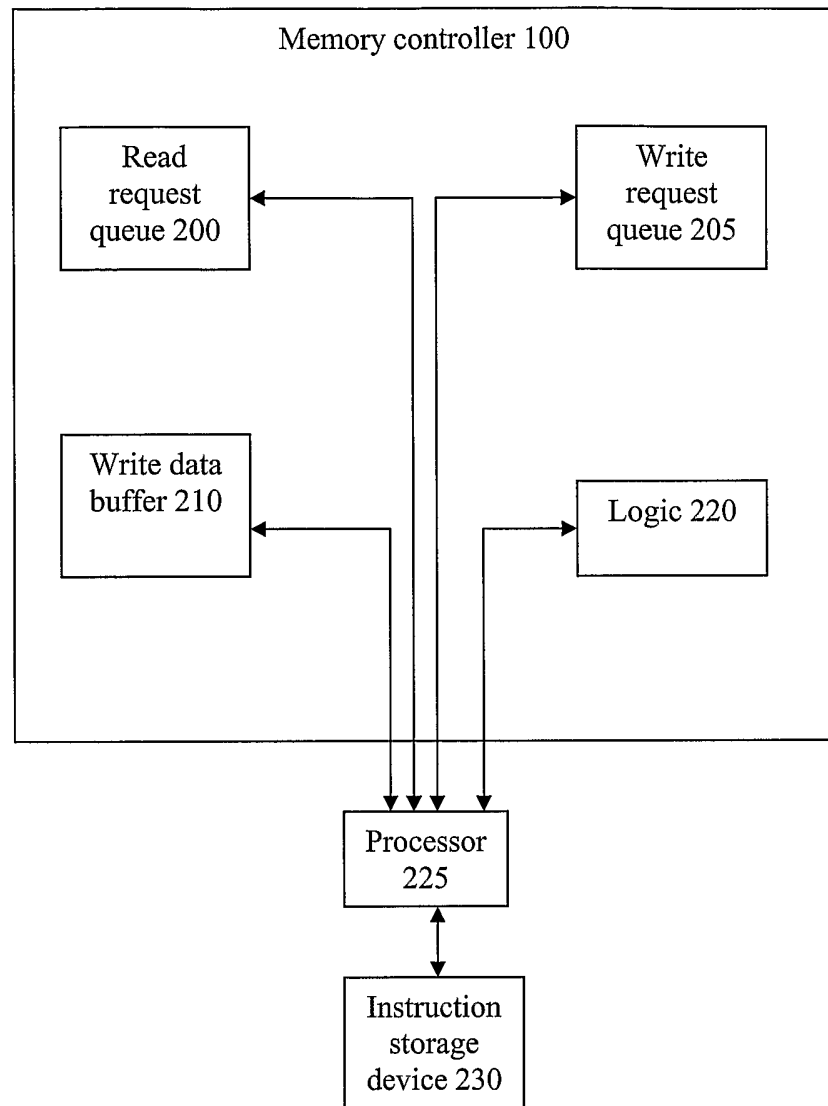


FIG. 2

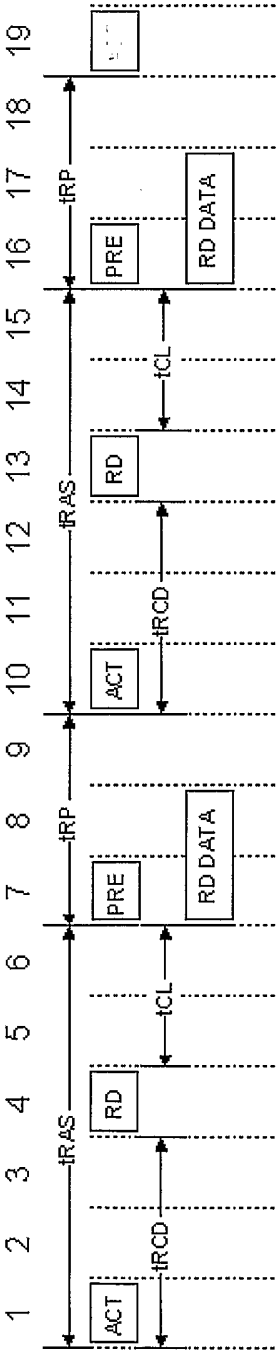


FIG. 3

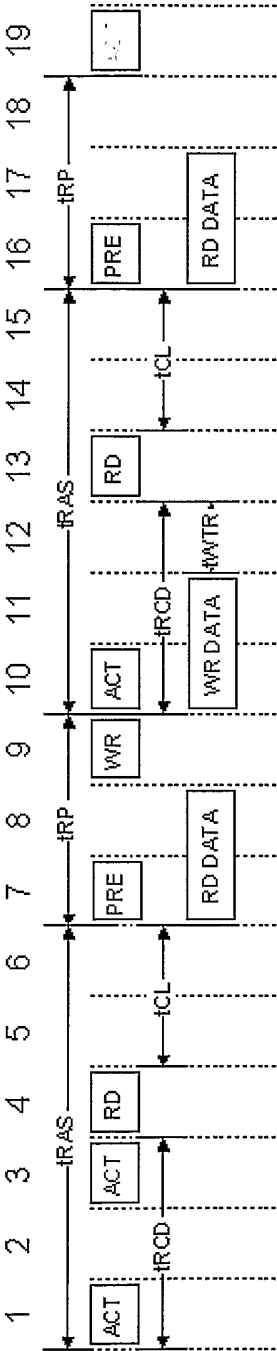


FIG. 4

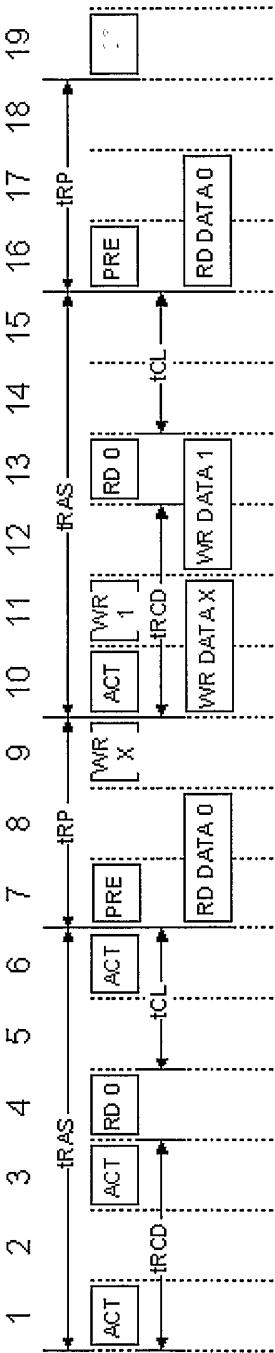


FIG. 5

5/7

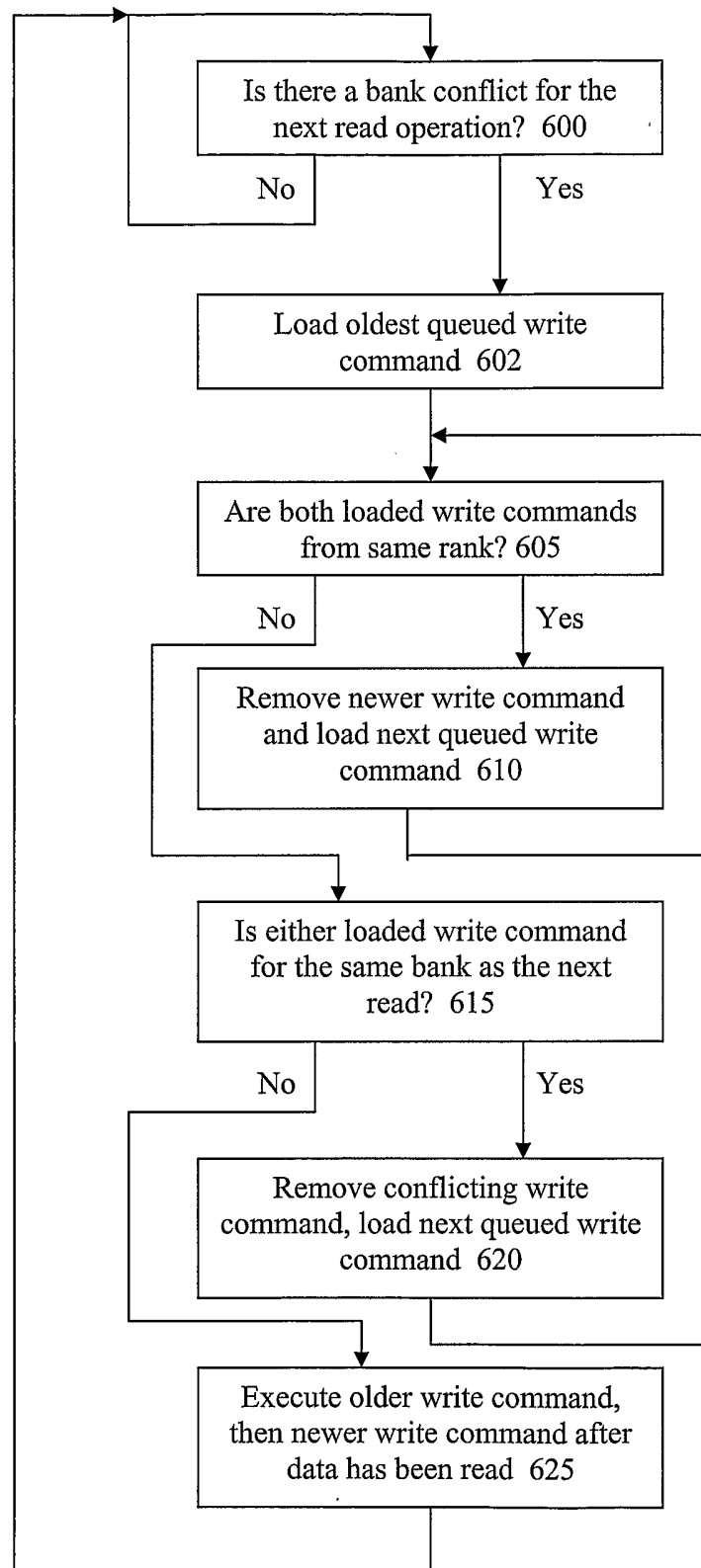


FIG. 6

6/7

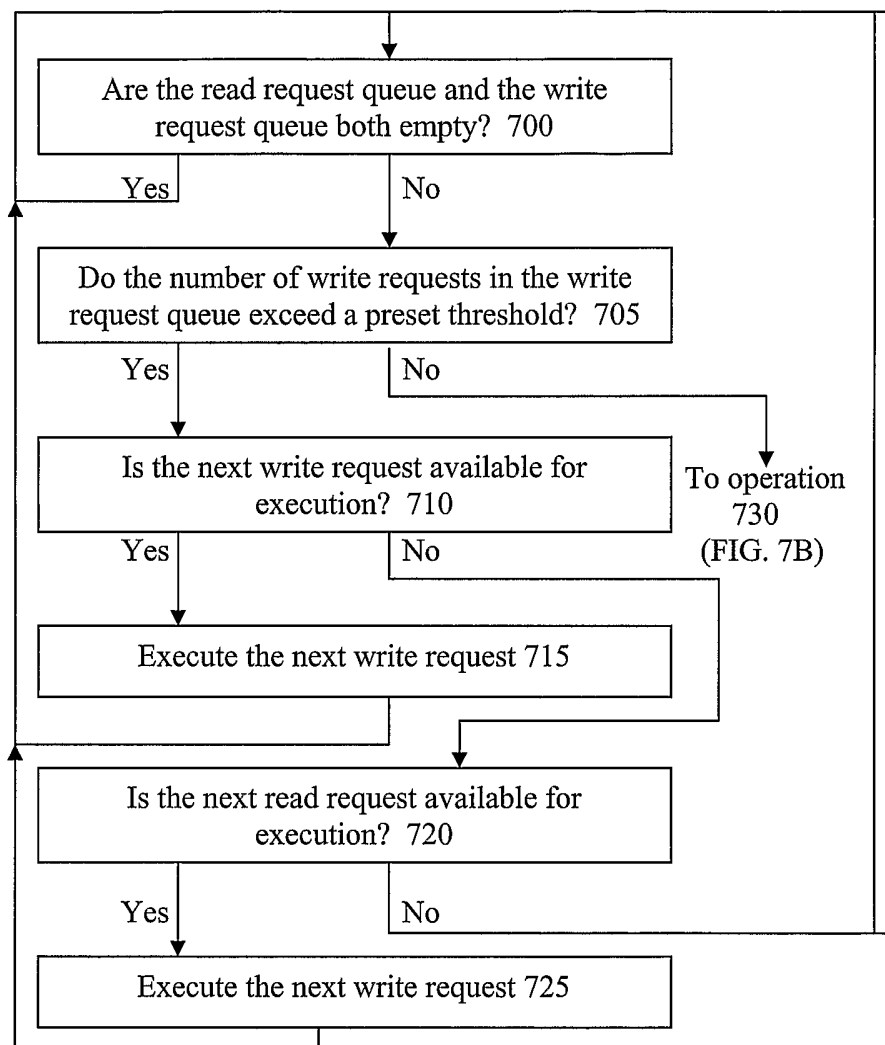


FIG. 7A

7/7

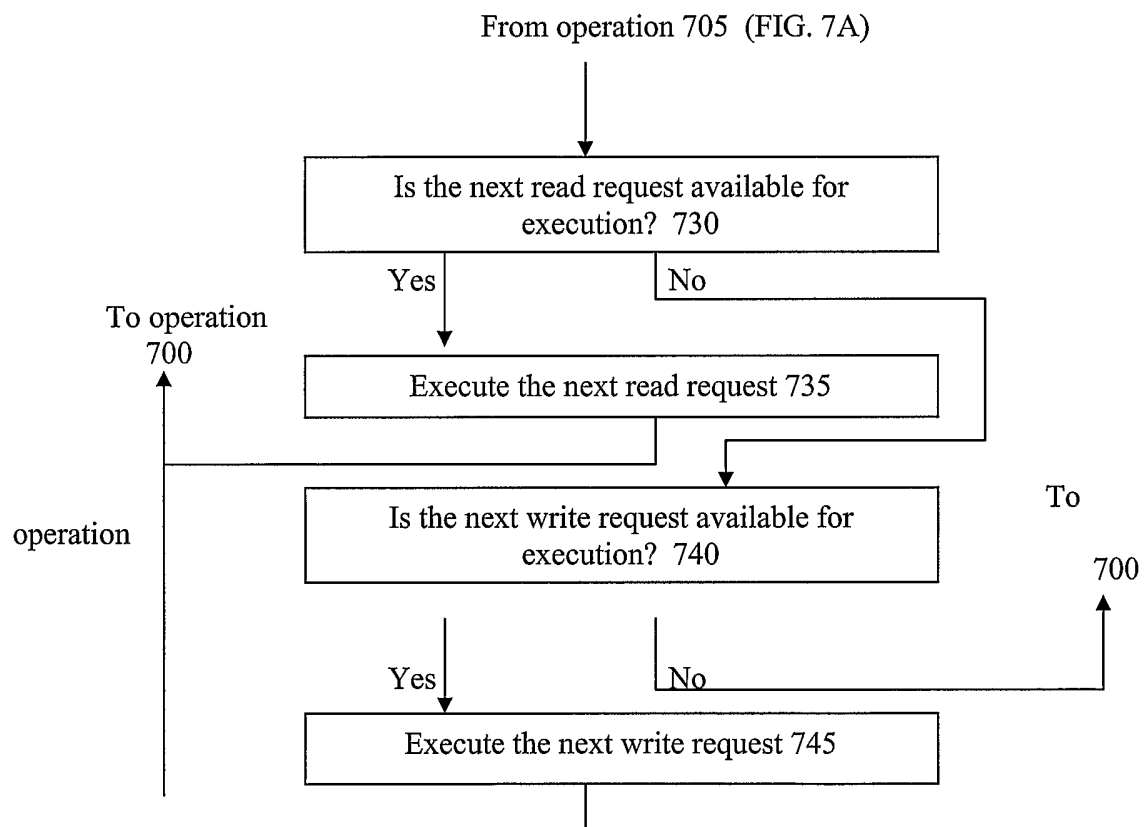


FIG. 7B