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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
4 January 2007 (04.01.2007)

PCT

(10) International Publication Number
WO 2007/002445 A2

(51) International Patent Classification: Not classified

(21) International Application Number:
PCT/US2006/024546

(22) International Filing Date: 23 June 2006 (23.06.2006)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
11/165,390 23 June 2005 (23.06.2005) US

(71) Applicant *(for all designated States except US)*: **INTEL CORPORATION** [US/US]; 2200 Mission College Boulevard, Santa Clara, California 95052 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **AKIYAMA, James** [US/US]; 15330 S.W. HERON COURT, BEAVERTON, Oregon 97007 (US). **CLIFFORD, William** [US/US]; 11223 67th NW, GIG HARBOR, Washington 98332 (US).

(74) Agents: **VINCENT, Lester, J.** et al.; BLAKELY SOKOLOFF TAYLOR & ZAFMAN, 12400 Wilshire Boulevard 7th Floor, Los Angeles, California 90025 (US).

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

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (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 "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WO 2007/002445 A2

(54) **Title:** MEMORY MICRO-TILING SPECULATIVE RETURNS

(57) **Abstract:** According to one embodiment, a memory controller is disclosed. The memory controller includes assignment logic and a transaction assembler. The assignment logic receives a request to access a memory channel and assigns the request to access one of two or more independently addressable subchannels within the channel. The transaction assembler combines the request with one or more additional requests to access the two or more independently addressable subchannels within the channel and facilitates a speculative return of data from a subchannel for which a subchannel request is not available.

MEMORY MICRO-TILING SPECULATIVE RETURNS

FIELD OF THE INVENTION

[0001] The present invention relates to computer systems; more particularly, the present invention relates to accessing memory control.

BACKGROUND

[0002] Computer systems implementing Unified Memory Architecture (UMA) feature a graphics controller that accesses main memory for video memory. However, the memory efficiency of UMA graphics systems may be limited due to CPU cache line size requirements. For example, the ideal memory access size for graphics may be 4 to 16 bytes, since graphics controllers can operate on one or a few pixels or texels at a time. Nevertheless, memory architectures are often optimized for the 64 byte CPU cache line size to optimize CPU memory efficiency. The result is that, on average, a significant amount of data read from memory may never used by the graphics controller. MANUFACTURERS OF DISCRETE GRAPHICS CONTROLLERS MINIMIZE THIS OVER FETCH BY USING NARROWER MEMORY CHANNELS. THIS SOLUTION, HOWEVER, IS NOT AVAILABLE FOR UMA-BASED INTEGRATED GRAPHICS CONTROLLERS.BRIEF DESCRIPTION OF THE DRAWINGS

[0003] The invention is illustrated by way of example and not limitation in the figures of the accompanying drawings, in which like references indicate similar elements, and in which:

[0004] Figure 1 is a block diagram of one embodiment of a computer system;

[0005] Figure 2 illustrates one embodiment of a memory controller

[0006] Figure 3 illustrates one embodiment of a logical virtual address;

[0007] Figure 4 illustrates another embodiment of a memory controller;

[0008] Figures 5A & 5B illustrate performance benefits; and

[0009] Figure 6 is a block diagram of another embodiment of a computer system.

DETAILED DESCRIPTION

[0010] Speculative returns for memory transactions are described. In the following detailed description of the present invention numerous specific details are set forth in order to provide a thorough understanding of the present invention. However, it will be apparent to one skilled in the art that the present invention may be practiced without these specific details. In other instances, well-known structures and devices are shown in block diagram form, rather than in detail, in order to avoid obscuring the present invention.

[0011] Reference in the specification to "one embodiment" or "an embodiment" means that a particular feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment of the invention. The appearances of the phrase "in one embodiment" in various places in the specification are not necessarily all referring to the same embodiment.

[0012] Figure 1 is a block diagram of one embodiment of a computer system 100. Computer system 100 includes a central processing unit (CPU) 102 coupled to an interface 105. In one embodiment, CPU 102 is a processor in the Pentium® family of Pentium® IV processors available from Intel Corporation of Santa Clara, California. Alternatively, other CPUs may be used. For instance, CPU 102 may be implemented using multiple processing cores. In yet other embodiments, computer system 100 may include multiple CPUs 102

[0013] In a further embodiment, a chipset 107 is also coupled to interface 105.

Chipset 107 includes a memory control component 110. Memory control component 110 may include a memory controller 112 that is coupled to a main system memory 115. Main system memory 115 stores data and sequences of instructions that are executed by CPU 102 or any other device included in system 100. In one embodiment, main system memory 115 includes dynamic random access memory (DRAM); however, main system memory 115 may be implemented using other memory types. Additional devices may also be coupled to interface 105, such as multiple CPUs and/or multiple system memories.

[0014] Memory control component 110 may be coupled to an input/output (I/O) control component 140 via an interface. I/O control component 140 provides an interface to I/O devices within computer system 100. I/O control component 140 may support standard I/O operations on I/O busses such as peripheral component interconnect (PCI) Express, accelerated graphics port (AGP), universal serial bus (USB), low pin count (LPC) bus, or any other kind of I/O bus (not shown).

[0015] According to one embodiment, graphics controller 160 is in communication with chipset 107 and is implemented to provide video graphics to a display monitor (not shown) coupled to computer system 100. Graphics controller 160 accesses main memory 115 for video memory. As discussed above, the memory efficiency of memory device supporting both a graphics system and a CPU is limited since memory access size for graphics is often ideally 4 to 16 bytes, while memory architectures are optimized for the 64 byte CPU line size to optimize CPU memory efficiency.

[0016] According to one embodiment, memory control component 110 reduces memory request size for graphics devices, while maintaining 64 byte memory transactions. A standard memory channel, such as based on DDR DRAM technology, has some physical width of m bits. A memory transaction is composed of T transfers for a

total logical width of M = m * T / 8 bytes. The bytes within each transaction are considered to have consecutive addresses. In subsequent discussion, the term width means the logical width.

[0017] Micro-Tiling breaks the M byte wide channel into S subchannels that are each N = M/S bytes wide and where N bytes of data are transferred on each subchannel. An address is presented to each subchannel, in which some number, I, of independent address bits may be different from corresponding bits in the addresses presented to the other subchannels. The data transferred on each subchannel is considered to represent a contiguous address range. However, the blocks of data on each subchannel are not necessarily from a contiguous address range. Each subchannel includes a subset of the total memory locations of the channel.

[0018] Figure 2 illustrates one embodiment of an memory control component 110 supporting Micro-Tiling. In one embodiment, a multi-channel memory subsystem has a Micro-Tiling memory controller per channel. Thus, as illustrated in Figure 2, memory control component 110 includes two memory controllers 112 (memory controllers 1 and 2), one for each of the two channels. Each channel includes S subchannels, each N bytes wide. Thus each channel is M=N*S bytes wide.

[0019] In this figure, requests to read or write memory are depicted as 2x2 arrays of squares possibly representing a 2x2 array of pixels or texels. Requests are shown before being assigned to a subchannel. After subchannel assignment, requests are numbered 0 - S-1 to suggest subchannel assignment. The N byte returns to requester 205 coupled to memory control component 110 occur in the case of a read transaction.

[0020] Memory control component 110 includes channel assignment logic 210 coupled to memory controllers 112. Channel assignment 210 assigns each request

received from requester 205 to a memory channel 240 via a memory controller 112. Further, each memory controller 112 includes subchannel assignment 215, reorder buffer 220 and transaction assembler 230. Thus, requests are assigned to memory controller 1 or memory controller 2 shown in Figure 2.

[0021] Sub-channel assignment 215 assigns each request to a subchannel within a memory channel 240. Reorder buffer 220 collects requests to enable transaction assembler 230 to attempt to assemble memory accesses for each memory 240 subchannel. According to one embodiment, each subchannel has an equal N byte width.

[0022] During operation of the system shown in Figure 2, a request to read or write a block of N bytes of data at address A enters a memory controller (1 or 2) is assigned to a subchannel and is placed in a reorder buffer 220. In one embodiment, the Identity Subchannel Assignment, s, is defined by the following process: the request address, A, is shifted right by $P = \log_2(N)$ bits, resulting in a new integer value Ã (e.g., Ã = A>>P); and s is the least significant $Q = \log_2(S)$ bits of Ã (e.g., s = Ã & ((1<<Q)-1)).

[0023] The memory controller forms a memory read transaction by selecting S read requests, one for each subchannel, from the reorder buffer 220. The memory controller forms a memory write transaction by selecting S write requests, one for each subchannel, from reorder buffer 220. The portion of the address represented by shared address lines is the same for all subchannel requests in the transaction.

[0024] Figure 3 illustrates one embodiment of an interpretation of address bits in a physical address. The choice of shared and independent address bits, and subchannel select bits shown in Figure 3 is for illustrative purposes since the division of the address bits above the P subchannel data address bits into shared and independent address bits, and subchannel select bits is arbitrary. The independent

address bits are different across subchannels, and are not necessarily contiguous. The address bits sent to a subchannel are the shared address bits and the independent address bits of that subchannel.

[0025] Figure 4 illustrates an embodiment of memory control component 110 assembling a 64 byte transaction from four 16 byte requests with only a single channel being shown. Figure 4 shows reorder buffer 220 implemented as a reorder queue for each subchannel. However, in other embodiments, reorder buffer 220 may be implemented via other mechanisms.

[0026] In this embodiment, transaction assembler 230 constructs a 64B memory request from 16B requests, one for each subchannel. All 16B requests forming the memory request have the same shared address bits. Thus assembler 230 looks into the queue for requests that can be assembled into a single transaction based upon whether requests have a common shared address.

[0027] Note that in the embodiment shown in Figure 4, assembler 230 cannot find a request for subchannel 1c. When attempting to form a transaction, the memory controller may not be able to find a request for each subchannel such that all have the same shared address segment (e.g., such that the value of each shared address bit is the same across all requests). A subchannel at which such a event occurs may be referred to as an "empty" subchannel.

[0028] According to one embodiment, if a subchannel cannot be filled by a request in the corresponding queue, the effect is that no transfer is performed from/to that subchannel. In such an embodiment, if a subchannel cannot be filled by a request an arbitrary location is read and the results are discarded. In an alternative embodiment, an additional control line is included per subchannel, which is used to power down a

subchannel when there is no corresponding request to that channel.

[0029] In yet another embodiment, speculative reads are performed and returned to a requester, rather than the retrieved data being discarded. A speculative return involves reading an N byte block of data on an otherwise unused subchannel and returning the data to some requester as for any other read request. Therefore, a memory controller may choose to read on the otherwise "empty" subchannel any aligned N byte block of data that has the same shared address bits as the data being accessed on the other subchannels in the same memory transaction, and making this data available for use by some requester.

[0030] According to one embodiment, a speculative return is implemented at transaction assembler 230 by assigning to the independent address lines associated with the "empty" subchannel, the independent address bits of a neighboring subchannel, resulting in two N byte blocks of data, having the same shared and independent address bits, being returned. Although described above as duplicating independent address lines, other embodiments may implement other mechanisms for choosing a block of data that is speculatively returned.

[0031] Referring to Figure 4 as an example, a request cannot be found for subchannel 1c. Thus, subchannel 1c is an "empty" subchannel. In the disclosed embodiment, transaction assembler 230 assigns the independent address bits of subchannel 1d to subchannel 1c and returns the data subchannel 1c to the requester associated with subchannel 1d. Thus, the speculative return improves effective memory utilization by performing speculative subchannel reads and returning the result to the requester. If this data is later needed by the requester, it may already/still be held, for example in a cache.

[0032] The Micro-Tiling memory subchannels can access discontiguous memory

addresses within some address ranges determined by the shared address bits and the I independent address bits. A judicious choice of I can therefore provide the increased concurrency and bandwidth efficiency of independent subchannels, balanced against the cost of duplicating I address signals to each subchannel.

[0033] Figures 5A & 5B illustrate performance benefits for Micro-Tiling. Each figure shows the rasterization of a triangle in a tiled address space, with each small square representing a 4 byte pixel or texel. Figure 5A shows overfetch in a standard memory system when requests are 64 bytes each. Each 4x4 block of pixels represents a 64 byte aligned block of memory. The triangle encompasses 57 pixels. With a standard memory subsystem, those 57 pixels are in 11 (64 byte) blocks of memory. Thus, in order to access those 57 pixels, an additional 119 pixels worth of data is accessed that may not be used (e.g., resulting in 32% efficiency).

[0034] Figure 5B shows the over fetch if requests are 16 bytes each and if all such requests can be utilized by the Micro-Tile Assembler to build 64 byte memory transactions with no unused subchannels. In this case, the triangle touches 23 2x2 pixel arrays, resulting in 35 additional pixels worth of data being accessed (e.g., resulting in 62% efficiency). The effectiveness of Micro-Tiling depends on the ability of the Assembler to construct fully populated memory transactions.

[0035] Figure 6 illustrates another embodiment of computer system 100. In this embodiment, chipset 107 includes a single control hub 620 as opposed to a separate memory control component and I/O control component. Consequently, memory controller 112 is included within CPU 102, with memory 115 being coupled to CPU 102. In such an embodiment, graphics controller 160 is coupled to control hub 620 and accesses main memory 115 via CPU 102

[0036] Whereas many alterations and modifications of the present invention will no doubt become apparent to a person of ordinary skill in the art after having read the foregoing description, it is to be understood that any particular embodiment shown and described by way of illustration is in no way intended to be considered limiting. Therefore, references to details of various embodiments are not intended to limit the scope of the claims, which in themselves recite only those features regarded as essential to the invention.

CLAIMS

What is claimed is:

1. A memory controller comprising:

assignment logic to receive a request to access a memory channel and to assign the request to access one of two or more subchannels within the channel; and

a transaction assembler to combine the request with one or more additional requests to access the two or more subchannels within the channel and to facilitate a speculative return of data from a subchannel for which a subchannel request is not available.

2. The memory controller of claim 1 wherein each of the subchannel requests include an independent address component and a shared address component.

3. The memory controller of claim 2 wherein the independent address component of a subchannel request is associated with a subchannel.

4. The memory controller of claim 3 wherein the transaction assembler facilitates the speculative return of data from a subchannel for which a request is not available, by selecting the independent address component to be associated with the subchannel for which a request is not be available.

5. The memory controller of claim 4 wherein the data speculatively read from the subchannel for which a request is not available, is returned to a requester.

6. The memory controller of claim 1 further comprising a reorder buffer to store the requests.

7. The memory controller of claim 6 wherein the reorder buffer includes a queue associated with each of the two or more subchannels.

8. The memory controller of claim 7 wherein each queue stores requests to be transferred to an associated subchannel.

9. A method comprising:

receiving a request at a memory controller to access a memory channel coupled to the memory controller;

assigning each of the requests to an associated independently addressable subchannel within the memory channel;

combining the request with one or more additional requests to access the two or more independently addressable subchannels within the channel; and

speculatively returning data from a subchannel for which a request is not available.

10. The method of claim 9 wherein the process of speculatively returning data from a subchannel for which a request is not available comprises selecting the independent address component to be associated with the subchannel for which a request is not be available.

11. The method of claim 9 further comprising storing the requests in a reorder buffer after assigning each of the requests to a subchannel.

12. The method of claim 9 further comprising forwarding the requests to the associated subchannels after assembling the requests.

13. The method of claim 10 further comprising returning the data speculatively returned from the subchannel for which a subchannel request is not be available to a requester.

14. A system comprising:

a memory device having one or more channels; and

a chipset, coupled to the memory device, having a memory controller to receive a request to access one of one or more memory channels, to assign the request to access one of two or more independently addressable subchannels within the channel, to combine the request with one or more additional requests to access the two or more independently addressable subchannels within the channel and to facilitate a speculative return of data from a subchannel for which a request is not available.

15. The system of claim 14 wherein comprises:

assignment logic to assign the request to access the subchannels; and

a transaction assembler to combine the requests and to facilitate the speculative return of data.

16. The system of claim 15 wherein the memory controller further comprises a reorder buffer to store the subchannel requests.

17. The system of claim 16 wherein the reorder buffer includes a queue associated with each of the two or more subchannels.

18. The system of claim 16 wherein the transaction assembler facilitates the speculative return of data from a subchannel for which a request is not available by

selecting an independent address component to be associated with the subchannel for which a request is not available.

19. The system of claim 14 further comprising a requester coupled to the memory device and the memory controller, wherein the data speculatively read from the subchannel for which a request is not available is returned to the requester.

20. The system of claim 14 wherein the chipset comprises a second memory controller to receive a request to access one of one or more memory channels and to assign the request to access one of two or more independently addressable subchannels within the channel.

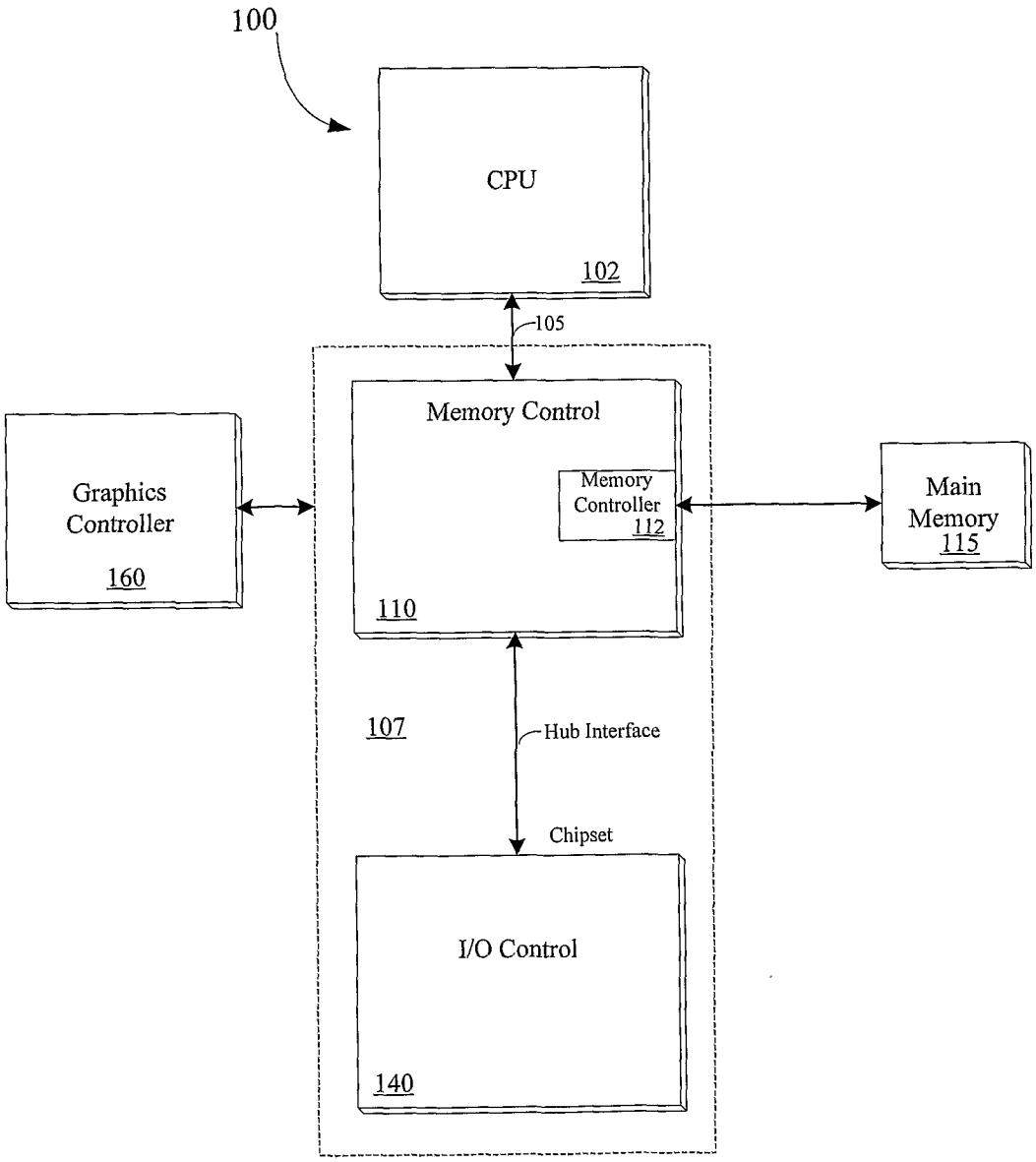
100
CPU
102
105
Memory Control
Memory Controller 112
Graphics Controller 160
Main Memory 115
110
107
Hub Interface
Chipset
I/O Control
140

Figure 1

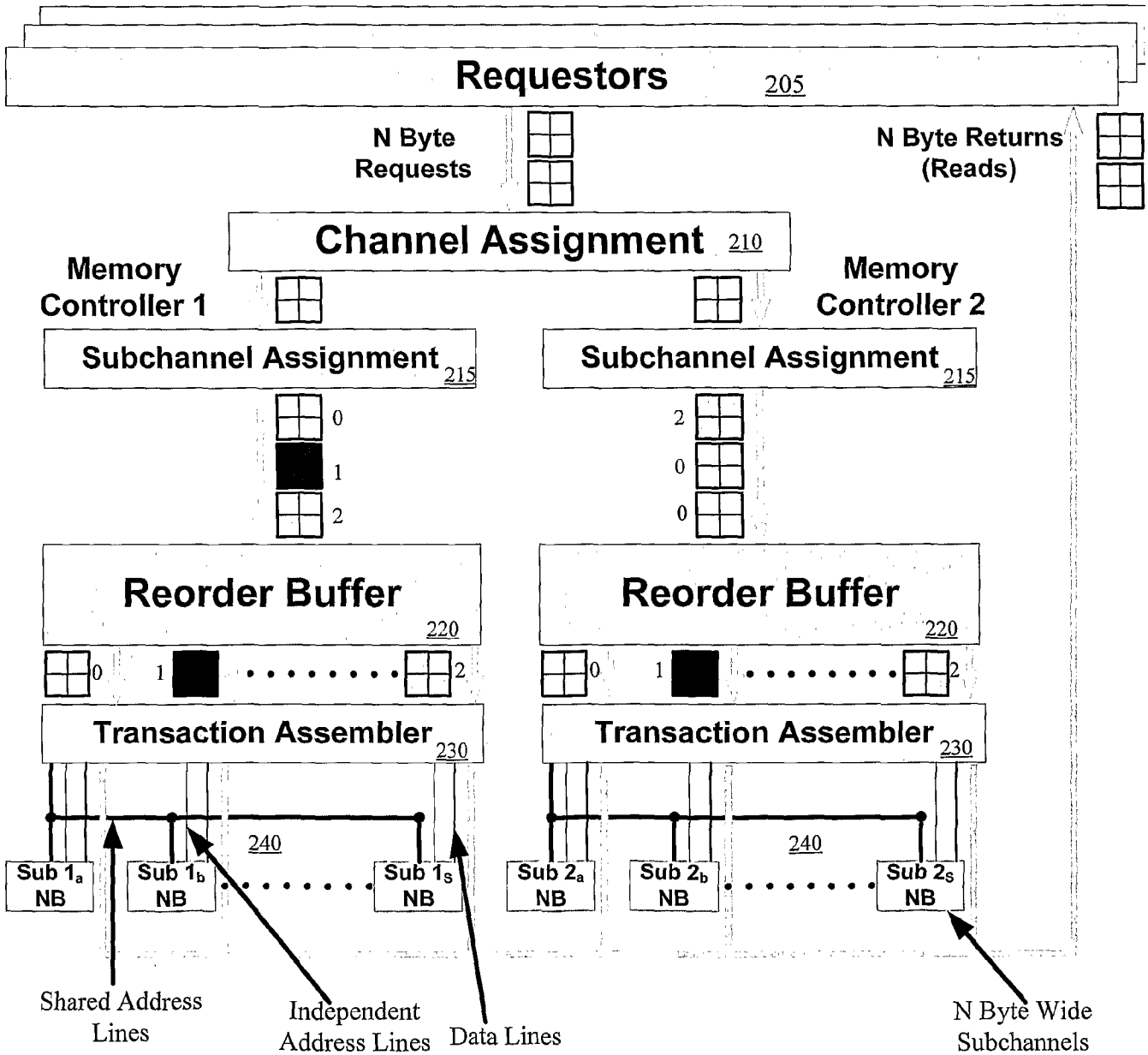
Requestors 205
N Byte Requests
N Byte Returns (Reads)
Channel Assignment 210
Memory Controller 1
Memory Controller 2
Subchannel Assignment 215
Subchannel Assignment 215
0
1
2
2
0
0
Reorder Buffer 220
Reorder Buffer 220
0 1 2
0 1 2
Transaction Assembler 230
Transaction Assembler 230
240
240
Sub 1a NB
Sub 1b NB
Sub 1s NB
Sub 2a NB
Sub 2b NB
Sub 2s NB
Shared Address Lines
Independent Address Lines
Data Lines
N Byte Wide Subchannels

Figure 2

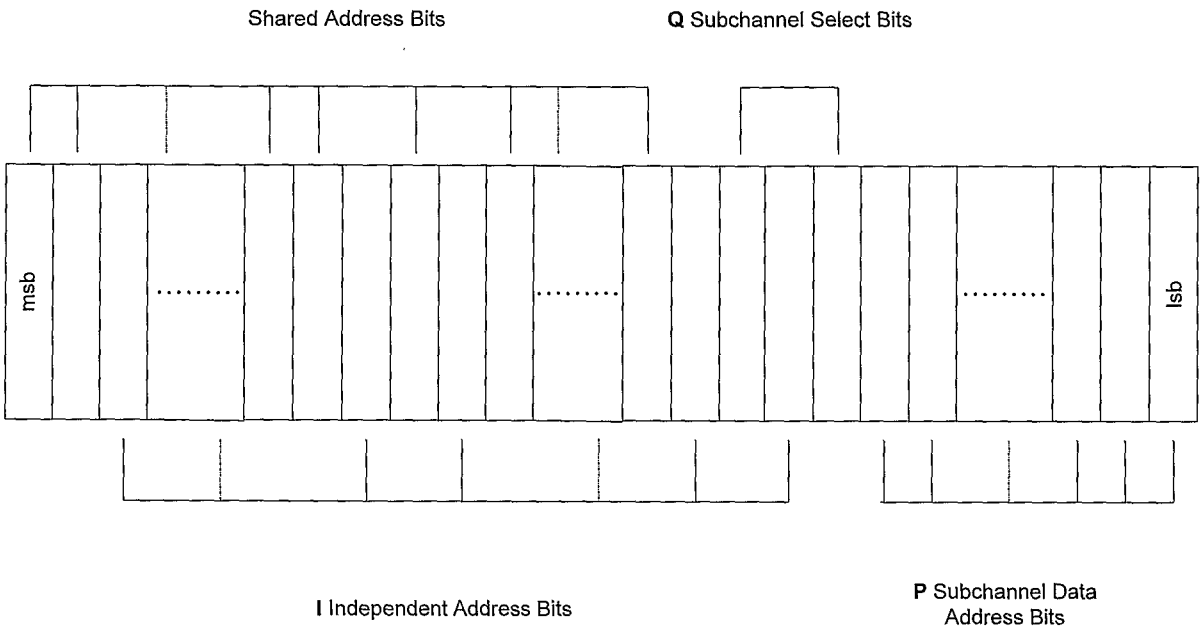
Shared Address Bits
Q Subchannel Select Bits
msb
lsb
I Independent Address Bits
P Subchannel Data Address Bits

Figure 3

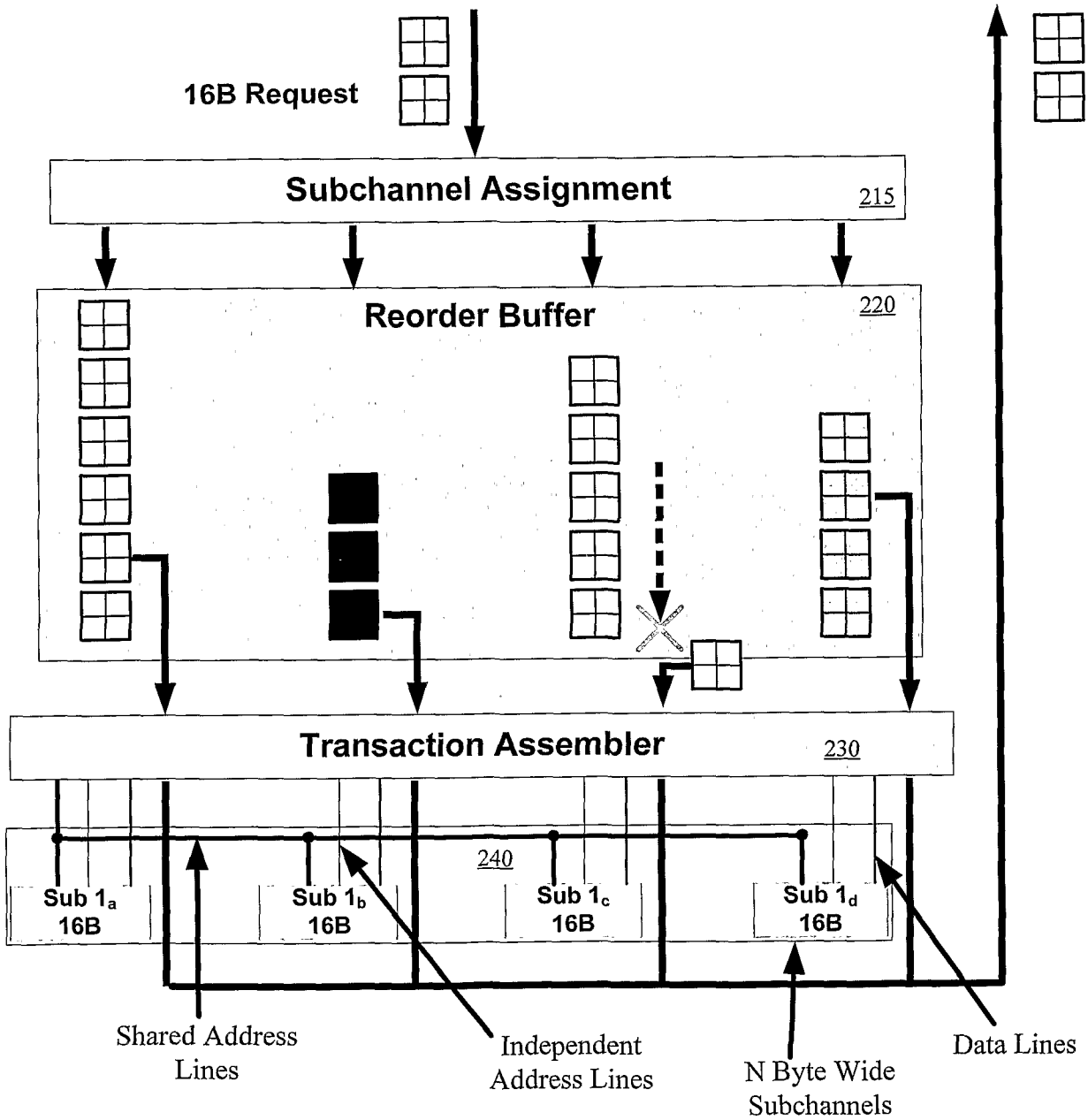
16B Request
Subchannel Assignment
215
Reorder Buffer
220
Transaction Assembler
230
240
Sub 1a 16B
Sub 1b 16B
Sub 1c 16B
Sub 1d 16B
Shared Address Lines
Independent Address Lines
N Byte Wide Subchannels
Data Lines

Figure 4

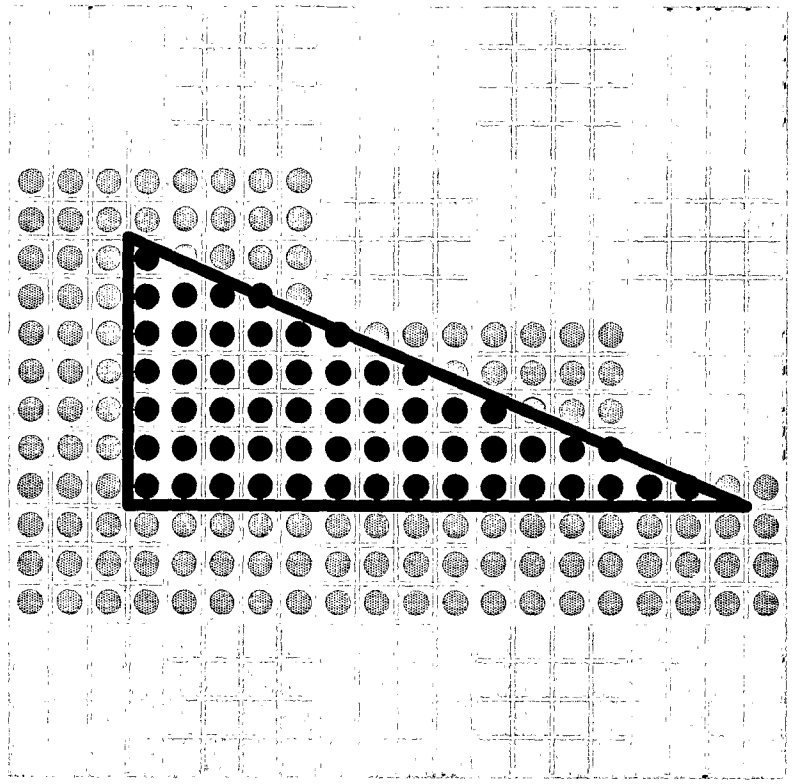

Figure 5A

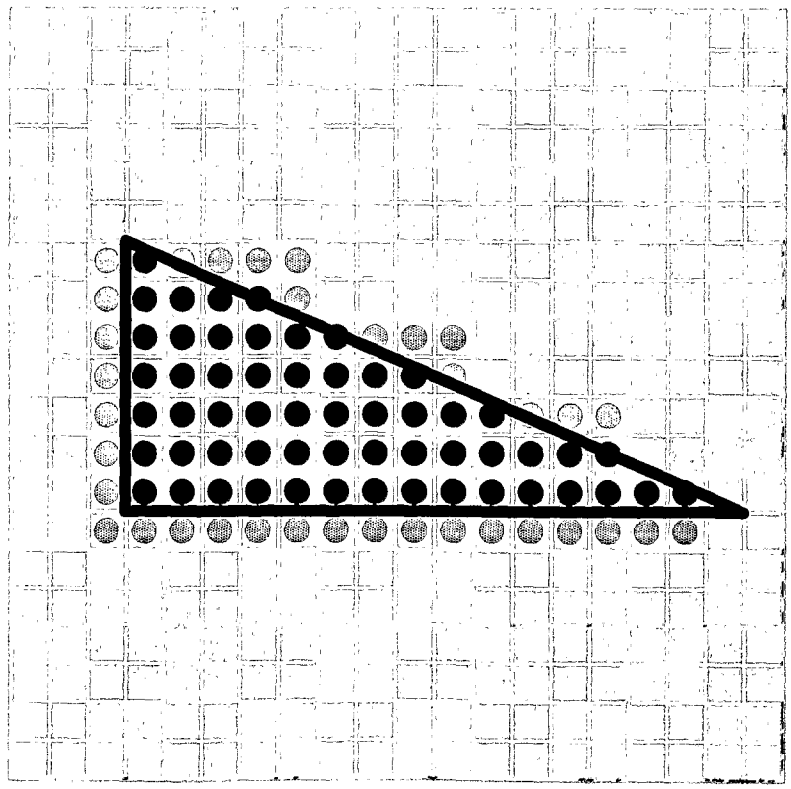

Figure 5B

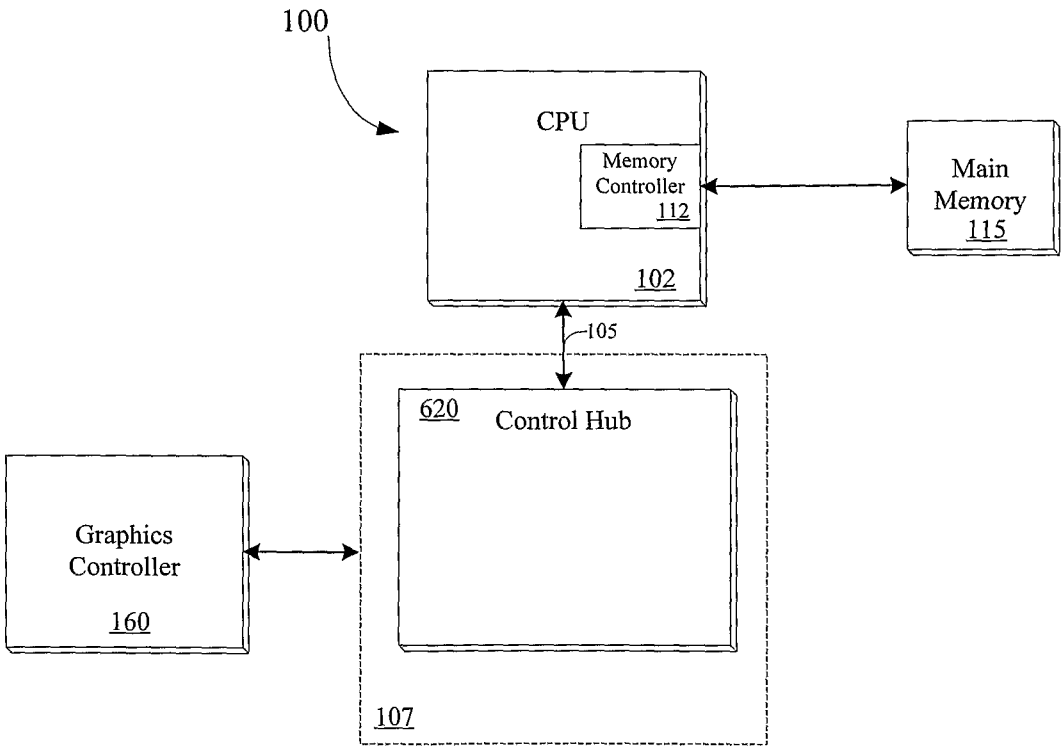
100
CPU
Memory Controller 112
102
Main Memory 115
105
620
Control Hub
Graphics Controller
160
107

Figure 6