



- (51) **International Patent Classification:**
G06F 13/14 (2006.01) *G06F 15/16* (2006.01)
G06F 13/38 (2006.01)
- (21) **International Application Number:**
PCT/US2012/035904
- (22) **International Filing Date:**
30 April 2012 (30.04.2012)
- (25) **Filing Language:** English
- (26) **Publication Language:** English
- (71) **Applicant (for all designated States except US):** **HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P.** [US/US]; 11445 Compaq Center Drive W., Houston, Texas 77070 (US).
- (72) **Inventors; and**
- (75) **Inventors/Applicants (for US only):** **GAITHER, Blaine, D.** [US/US]; 3404 E. Harmony Road, Fort Collins, Colorado 80528 (US). **BROOKS, Robert, J.** [US/US]; 3404 E. Harmony Road, Fort Collins, Colorado 80528 (US). **OSECKY, Benjamin, D.** [US/US]; 3404 E. Harmony Road, Fort Collins, Colorado 80528 (US). **EVERTSON, Kathryn, A.** [US/US]; 3404 E. Harmony Road, Fort Collins, Colorado 80528 (US). **WHEELER, Andrew, R.** [US/US]; 3404 E. Harmony Road, Fort Collins, Colorado 80528

(US). **FISK, David** [US/US]; 1925 West John Carpenter Freeway, Suites 400/475, Irving, Texas 75063 (US).

- (74) **Agents:** **MCDOWELL, Robert** et al.; Hewlett-Packard Company, Intellectual Property Administration, 3404 E. Harmony Road, Mail Stop 35, Fort Collins, CO 80528 (US).

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

- (84) **Designated States** (*unless otherwise indicated, for every kind of regional protection available*): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, 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,

[Continued on next page]

- (54) **Title:** REFLECTIVE MEMORY BRIDGE FOR EXTERNAL COMPUTING NODES

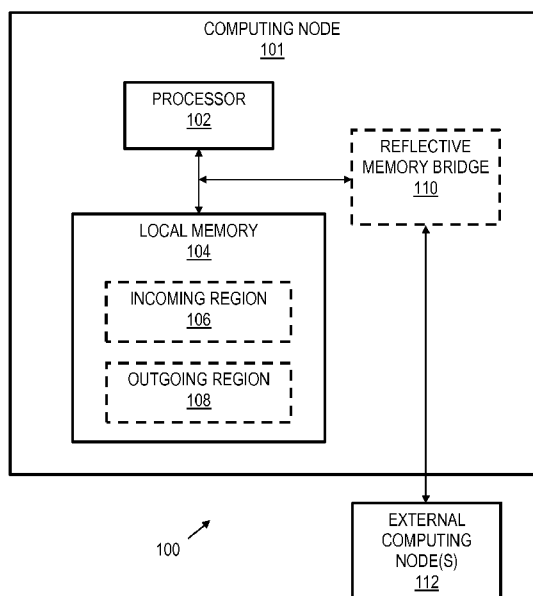


FIG. 1

- (57) **Abstract:** In at least some examples, a computing node includes a processor and a local memory coupled to the processor. The computing node also includes a reflective memory bridge coupled to the processor. The reflective memory bridge maps to an incoming region of the local memory assigned to at least one external computing node and maps to an outgoing region of the local memory assigned to at least one external computing node.



TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, **Published:**
ML, MR, NE, SN, TD, TG).

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

Declarations under Rule 4.17:

— *as to the identity of the inventor (Rule 4.17(i))*

REFLECTIVE MEMORY BRIDGE FOR EXTERNAL COMPUTING NODES

BACKGROUND

[0001] Database systems are evolving from disk-based single server systems with high input/output (I/O) rates and small memories, to clusters of independent nodes which hold the database entirely in memory. These systems often rely on horizontal partitioning (or sharing) for scaling and k-safety (synchronous replication) to provide durability.

[0002] Data contention and metadata contention are ongoing challenges for online transaction processing (OLTP) systems. The probability of contention increases as the number of parallel transactions in the system increases. Transaction response time is linked to the number of active parallel transactions by Little's law. This law states that the number of transactions in a system equals the throughput multiplied by the response time.

BRIEF DESCRIPTION OF THE DRAWINGS

[0003] For a detailed description of illustrative examples of the disclosure, reference will now be made to the accompanying drawings in which:

[0004] Figure 1 shows an example of a computing node system in accordance with the disclosure;

[0005] Figures 2A-2D show other examples of computing node systems in accordance with the disclosure;

[0006] Figure 3 shows an example of an integrated circuit in accordance with the disclosure;

[0007] Figure 4 shows an example of a monitoring system for a computing node in accordance with the disclosure;

[0008] Figure 5 shows an example of another monitoring system for a computing node in accordance with the disclosure;

[0009] Figure 6 shows an example of a computer system in accordance with the disclosure; and

[0010] Figure 7 shows an example of a method in accordance with the disclosure.

NOTATION AND NOMENCLATURE

[0011] Certain terms are used throughout the following description and claims to refer to particular system components. As one skilled in the art will appreciate, computer companies may refer to a component by different names. This document does not intend to distinguish between components that differ in name but not function. In the following discussion and in the claims, the terms “including” and “comprising” are used in an open-ended fashion, and thus should be interpreted to mean “including, but not limited to...” Also, the term “couple” or “couples” is intended to mean either an indirect, direct, optical or wireless electrical connection. Thus, if a first device couples to a second device, that connection may be through a direct electrical connection, through an indirect electrical connection via other devices and connections, through an optical electrical connection, or through a wireless electrical connection.

DETAILED DESCRIPTION

[0012] The following discussion is directed to methods and systems for employing reflective memory in a multi-node computing and database environment. In one example, a computing node may include local memory and a reflective memory bridge (controller) that maps to an incoming region of the local memory assigned to at least one external computing node and that maps to an outgoing region of the local memory assigned to at least one external computing node. The reflective memory bridge may be integrated with another component such as a processor, a node controller, a coherency bus, or a peripheral component interconnect (PCI) card. As used herein, an “external computing node” refers to a computer running a different instance of an operating system and not necessarily running the same operating system or using the same CPU architecture. As used herein, “reflective memory” refers to a mechanism for snarfing data written to one or more regions of memory of a node. In the disclosed examples, when a store falls within a particular range of addresses, a copy of the data is forwarded to at least one external computing node. The external node’s reflective memory controller stores the write data in that node’s memory at a relative memory location that corresponds to the particular range from which the data was sent.

[0013] In the disclosed examples, reflective memory is used to maintain a coherent copy of a memory region in one node or multiple nodes in order to maintain the database concept of k-safety. Examples of applications which use k-safety are Vertica and Hadoop. The disclosed examples maintain the concept of k-safety with minimum latency and without requiring input/output path instructions to be executed on the nodes involved. The interconnect between nodes does not have to be a reflective memory bus, but rather can be any interconnect including a network. The network should maintain the order of forwarded stores from each node-region pair to each particular node. No other ordering is necessary (e.g., no ordering between regions is needed).

[0014] Without limitation to other examples, the reflective memory bridge may specify for an outgoing region: a name, addresses associated with the outgoing region, and a set of external computing nodes that are to receive updates to the outgoing region. Further, the reflective memory bridge may specify for an incoming region: a name, addresses associated with the incoming region, and a set of external computing nodes that are to reflect to the incoming region. In some examples, a reflective memory bridge may specify a plurality of different outgoing regions and/or incoming regions. Further, a reflective memory bridge may update the plurality of different outgoing regions and/or incoming regions upon request from a user or administrator. Further, a reflective memory bridge may be part of with a coherency fabric and/or an interconnect monitoring interface of a computing node.

[0015] In some examples, the addresses corresponding to the incoming regions and outgoing regions are physical memory addresses. In other examples, snarfing may occur pre-V2P (virtual-to-physical), and a field in the virtual address would contain the remote node ID or a broadcast key. Likewise, an incoming transaction would have a virtual address which would get protection-checked and translated by the receiving core just like a store that originated from its own pipeline.

[0016] In some examples, a computing node includes a database with access to the reflective memory bridge. Without limitation to other examples, the database may utilize the reflective memory bridge to perform operations such as

communicating commands or results to at least one external computing node, distributing logs, updates, and new data to at least one external computing node, communicating information for two-phase commits to at least one other computing node, maintaining k-safety by reflecting at least part of a projection being written to a mirror node, and/or writing to a logging appliance.

[0017] The disclosed examples for employing reflective memory in a multi-node computing and database environment differ from message-based systems that transmit messages with a contiguous buffer over a network, or systems that rely on a shared memory paradigm. While the shared memory paradigm is easy to program and is efficient, shared memory systems are subject to single points of failure that can bring down all systems accessing the shared memory region. Meanwhile, message-based systems require buffers to be constructed after which the message is sent to the other system. Because no part of the message is sent until the buffer filling is completed, significant instruction path length increases may occur during I/O operations and may interrupt processing. Use of reflective memory operations to reduce transaction response time across multiple computing nodes as disclosed reduces lock and latch related contention and in the end helps to further reduce response time.

[0018] The disclosed examples for employing reflective memory in a multi-node computing and database environment should not be interpreted, or otherwise used, as limiting the scope of the disclosure, including the claims. In addition, one skilled in the art will understand that the following description has broad application, and the discussion of any particular example is not intended to intimate that the scope of the disclosure, including the claims, is limited to that example.

[0019] Figure 1 shows an example of a computing node system 100 in accordance with the disclosure. As shown, the computing node system 100 comprises a computing node 101 in communication with external computing nodes 112. More specifically, the computing node 101 comprises a processor 102 coupled to a local memory 104 with an incoming region 106 and an outgoing region 108 associated with a reflective memory bridge 110. The reflective memory bridge 110 maps to the incoming region 106, which is assigned to at

least one external computing node 112. The reflective memory bridge 110 also maps to the outgoing region 106, which is assigned to at least one external computing node 112. The external computing nodes 112 corresponding to the incoming region 106 may be the same or may be different than the external computing nodes 112 corresponding to the outgoing region 108. Although not required, the computing node 101 may optionally include input/output (I/O) interface components (not shown for convenience).

[0020] For the incoming region 106, the reflective memory bridge 110 specifies a name or identifier, physical addresses associated with the incoming region 106, and a set of external computing nodes 112 that are to reflect to the incoming region 106. For the outgoing region 108, the reflective memory bridge 110 specifies a name or identifier, physical addresses associated with the outgoing region 108, and a set of external computing nodes that are to receive updates to the outgoing region 108. In at least some examples, stores from the reflective memory bridge 110 are not reflected. More specifically, stores generated by a destination node will not be reflected. Thus, two nodes are able to reflect the same memory locations to each other (similar to shared memory) without risking an infinite loop. In other examples, regions are set up so that node A reflects to node B, and node B reflects the region to node C. In another example, node A reflects to nodes B and C.

[0021] In some examples, the reflective memory bridge 110 corresponds to an application-specific integrated circuit (ASIC) configured to perform the reflective memory operations described herein. Alternatively, the reflective memory bridge 110 may correspond to a processor executing reflective memory bridge instructions stored in a memory available to the processor. Alternatively, the reflective memory bridge 110 may correspond to programmable logic configured to perform the reflective memory operations described herein. The decisions regarding how to implement the reflective memory bridge may be based, for example, on performance criteria, cost criteria, and/or compatibility criteria. For all implementations, the reflective memory bridge 110 needs to be compatible with an existing or planned coherency fabric of a computing node.

[0022] In operation, the reflective memory bridge 110 observes all stores to the outgoing region 108 and transmits those stores to others of the external computing nodes 112. The external computing nodes 112 may each comprise their own local reflective memory bridge to repeat those stores to local memory. In this manner, a message or change in state is transmitted as it is composed rather than waiting to transmit the first byte until the last byte is copied into a buffer as is done for buffer-based systems. Also, there is no requirement that the words in a message be written in order. To be clear, while there is no requirement that the words in a region be written in any particular order, the words written from a node to the region will be reflected in order to all sharing nodes. If two nodes are writing to the same region all sharing nodes may not see those writes in the same order. It is the responsibility of reflective memory bridge software to deal with this possibility. Single writer multiple readers are able to see stores in the same order.

[0023] Thus, while traditional message passing waits to begin transmission until the last store to a buffer is completed, reflective memory operations implemented by the reflective memory bridge 110 allows each store to be sent immediately (i.e., message latency is reduced).

[0024] The disclosed approach differs from traditional sender-managed communications (e.g. remote direct memory access or RDMA) in that there is no need to build a buffer, nor execute a protocol stack, nor field an interrupt (these operations may still be optionally done). This permits flexible zero copy solutions.

[0025] In the disclosed computing node system 100, no memory is shared and there are no multi-node single points of failure. In other words, each computing node (e.g., computing nodes 101 and 112) has its own local memory into which updates are copied, and none of the computing nodes 101 and 112 reads another's local memory. Readers and writers may fail, but this will not cause a hardware crash in other nodes. Also, since a region of memory is watched for update, it is possible to keep data structures update in situ, and not require a buffer to be created. This allows scatter-scatter communication. Although only one incoming region 106 and one outgoing region 108 are shown in Figure 1, additional incoming regions and/or outgoing regions may be defined, where each

region can independently be reflected to a different set of servers corresponding to the external computing nodes 112.

[0026] To set up reflective memory operation, each computing node (e.g., computing nodes 101 and 112) of the computing node system 100 may communicate with its respective reflective memory bridge through a memory mapped region. This region is used to set up one or more outgoing memory regions to be reflected. In some examples, for each outgoing region the following is specified: a name for the region (if empty this region is not active), physical addresses that contribute to the region, and a set of nodes that will receive updates to this region. Similarly, for each incoming region, the following may be specified: a name for the region (matched to incoming regions or empty if this region is not active), physical addresses that comprise the region, and a set of nodes which may reflect to this region (the nodes may be checked to reduce errors). The specified incoming regions will receive data reflected from a sender node, where the mapping of addresses is relative to the first byte of each region.

[0027] Additional extensions to the computing node system 100 may include at least one memory mapped register that tracks a count of how many writes of a region are still in hardware buffers and have not yet been transferred. The memory mapped register(s) is incremented whenever a write is observed and decremented when a write has been sent to the external computing nodes 112. In this manner, a sender can be sure that a message has been sent. Along the same lines, an interrupt could be raised if the hardware buffers have reached a threshold fill level, indicating a possible interconnect problem.

[0028] For some examples of the computing node system 100, there may be a need for safe and efficient synchronization. Toward that end, memory mapped interfaces could be used to allow remote read/exchange constructs and/or read/increment constructs with timeout/error indications in the event that the memory is unavailable. In some examples, one of the Lamport algorithms may be used for high-priority sections. Without limitation to other examples, the reflective memory bridge 110 may enable the computing node 101 to communicate commands and results to at least one of the external computing nodes 112, to distribute logs, updates and new data across the external

computing nodes 112, to communicate information for two-phase commits to at least one of the external nodes 112, to maintain k-safety by reflecting the part of a projection that is being written to at least at least one of the external computing nodes 112 operating as a mirror node for the computing node 101, and to write to a logging appliance corresponding to at least one of the external computing nodes 112.

[0029] Use of the reflective memory bridge as described herein reduces communications related latencies significantly and requires no driver overhead. Thus, there is little to no impact on instruction path length and building of a contiguous buffer is avoided. Further, zero copy is possible. Further, multi-node single points of failure are avoided. Further, in situ synchronization of data structures is possible. Further, by reducing communications latency the disclosed reflective memory operations enable the highest scaling of horizontally partitioned databases.

[0030] Figures 2A-2D show other examples of computing node systems 200A-200D in accordance with the disclosure. As shown, in the computing node system 200A of Figure 2A, the computing node 201A comprises a processor 202 with the reflective memory bridge 110 integrated therein. In the computing node 201A, a coherency fabric 212 enables reflective memory operations involving the processor 202, the local memory 104, and the external computing nodes 112. Alternatively, the reflective memory bridge 110 may be implemented off of the coherency fabric 212 or directly from the bridge, or from the bridge to an I/O device to an interconnect that is not the coherency fabric 212.

[0031] Further, a database 204 of the computing node 201A is able to request that the reflected memory controller 110 perform reflective memory operations to communicate commands and results to at least one of the external computing nodes 112, to distribute logs, updates and new data across the external computing nodes 112, to communicate information for two-phase commits to at least one of the external nodes 112, to maintain k-safety by reflecting the part of a projection that is being written to at least at least one of the external computing nodes 112 operating as a mirror node for the computing node 201A, and to write

to a logging appliance corresponding to at least one of the external computing nodes 112.

[0032] In the computing node system 200B of Figure 2B, the computing node 201B comprises the processor 102 and a coherency fabric 222 with the reflective memory bridge 110 integrated therein. In the computing node 201B, the coherency fabric 222 enables reflective memory operations involving the processor 102, the local memory 104, and the external computing nodes 112. Further, the database 204 of the computing node 201B is able to request that the reflected memory controller 110 perform reflective memory operations to communicate data, logs, commands, two-phase commits, and/or k-safety mirroring projections to at least one of the external computing nodes 112 as described herein.

[0033] In the computing node system 200C of Figure 2C, the computing node 201C comprises the processor 102 and a coherency fabric 232 with a node controller 234 having the reflective memory bridge 110 integrated therein. Without limitation to other examples, the node controller 234 may operate in accordance with a quickpath interconnect (QPI) architecture. In the computing node 201C, the coherency fabric 232 enables reflective memory operations involving the processor 102, the local memory 104, and the external computing nodes 112. Further, the database 204 of the computing node 201C is able to request that the reflected memory controller 110 perform reflective memory operations to communicate data, logs, commands, two-phase commits, and/or k-safety mirroring projections to at least one of the external computing nodes 112 as described herein.

[0034] In the computing node system 200D of Figure 2D, the computing node 201D comprises the processor 102 and a coherency fabric 242 with a PCI card 244 having the reflective memory bridge 110 integrated therein. In the computing node 201D, the coherency fabric 232 enables reflective memory operations involving the processor 102, the local memory 104, and external computing nodes 112. Further, the database 204 of the computing node 201D is able to request that the reflected memory controller 110 perform reflective memory operations to communicate data, logs, commands, two-phase commits, and/or k-

safety mirroring projections to at least one of the external computing nodes 112 as described herein.

[0035] In the computing node systems 200A-200D of Figures 2A-2D, the external computing nodes 112 may implement any of the reflective memory bridge configurations described herein. The position of the reflective memory bridge 110 for a particular computing node (e.g., computing node 101, computing nodes 201A-201D, and computing nodes 112) may vary according to considerations such as implementation timing, implementation complexity, spacing requirements, and/or component availability.

[0036] Figure 3 shows an example of an integrated circuit 300 in accordance with the disclosure. The integrated circuit 300 may correspond to, for example, a processor, a node controller, a PCI card, or a coherency fabric component. As shown, the integrated circuit 300 comprises the reflective memory bridge 110, which is shown to specify for each incoming region: a name for the region (matched to incoming regions or empty if this region is not active), physical addresses that comprise the region, and a set of nodes which may reflect to this region (the nodes may be checked to reduce errors). Further, the reflective memory bridge 110 of the integrated circuit 300 may specify for each outgoing region: a name for the region (if empty this region is not active), physical addresses that contribute to the region, and a set of nodes that will receive updates to this region. As needed, the incoming regions or outgoing regions specified for the reflective memory bridge 110 of the integrated circuit 300 may be updated by a user or administrator.

[0037] Figure 4 shows an example of a monitoring system 400 for a computing node in accordance with the disclosure. The monitoring system 400 is able to detect when an interconnect for reflective memory operations becomes unavailable. As shown, the monitoring system 400 may comprise a buffer 402 that receives and releases writes of an outgoing region (e.g., the outgoing region 108) of local memory assigned for reflective memory operations. The monitoring system 400 also comprises a memory mapped register 404 that maintains a count 406 based on incrementing the count 406 when the buffer 402 receives a write from an outgoing region and based on decrementing the count 406 when

the buffer 402 releases a write for transmission to an external computer node. If the count 406 exceeds a predetermined threshold (e.g., a fill level for the buffer 402 exceeds a predetermined threshold), an interrupt or notification is sent from the memory mapped register 404 to an interconnect controller 408 to report an unavailable interconnect.

[0038] Figure 5 shows an example of another monitoring system 500 for a computing node in accordance with the disclosure. The monitoring system 500 is able to detect when a memory region for reflective memory operations becomes unavailable. As shown, the monitoring system 500 may comprise a memory mapped interface 504 with timeout/error logic 506 to assert a timeout or error based on the availability information. The memory mapped interface 504 may, for example, support remote read/exchange constructs or remote read/increment constructs with timeout or error indications provided by the timeout/error logic 510 when an outgoing region or an incoming region for reflective memory operations is unavailable. In such case, an interrupt or notification is sent from the memory mapped interface 504 to an interconnect controller 508 to report an unavailable memory region. The unavailability of a memory region or interconnect may occur for several reasons. For example, the receiver might crash causing the buffer to fill, an interconnect may have a glitch causing it to stop communications with an external node or nodes, or human error, (e.g., someone kicks out a power or network cable) may cause a memory region or interconnect to be unavailable.

[0039] Figure 6 shows an example of various components of a computer system or node 600 in accordance with the disclosure. The computer system 600 may perform various operations to support the reflective memory operations described herein. The computer system 600 may correspond to components of the computing node 101, components of the computing node 201A-201D, or components of the external computing nodes 112 described herein.

[0040] As shown, the computer system 600 includes a processor 602 (which may be referred to as a central processor unit or CPU) that is in communication with memory devices including secondary storage 604, read only memory (ROM) 606, random access memory (RAM) 608, input/output (I/O) devices 610, and

network connectivity devices 612. The processor 602 may be implemented as one or more CPU chips.

[0041] It is understood that by programming and/or loading executable instructions onto the computer system 600, at least one of the CPU 602, the RAM 608, and the ROM 606 are changed, transforming the computer system 600 in part into a particular machine or apparatus having the novel functionality taught by the present disclosure. In the electrical engineering and software engineering arts functionality that can be implemented by loading executable software into a computer can be converted to a hardware implementation by well-known design rules. Decisions between implementing a concept in software versus hardware typically hinge on considerations of stability of the design and numbers of units to be produced rather than any issues involved in translating from the software domain to the hardware domain. For example, a design that is still subject to frequent change may be implemented in software, because re-spinning a hardware implementation is more expensive than re-spinning a software design. Meanwhile, a design that is stable that will be produced in large volume may be preferred to be implemented in hardware, for example in an application specific integrated circuit (ASIC), because for large production runs the hardware implementation may be less expensive than the software implementation. Often a design may be developed and tested in a software form and later transformed, by well-known design rules, to an equivalent hardware implementation in an application specific integrated circuit that hardwires the instructions of the software. In the same manner as a machine controlled by a new ASIC is a particular machine or apparatus, likewise a computer that has been programmed and/or loaded with executable instructions may be viewed as a particular machine or apparatus.

[0042] The secondary storage 604 may be comprised of one or more disk drives or tape drives and is used for non-volatile storage of data and as an overflow data storage device if RAM 608 is not large enough to hold all working data. Secondary storage 604 may be used to store programs which are loaded into RAM 608 when such programs are selected for execution. The ROM 606 is used to store instructions and perhaps data which are read during program execution.

ROM 606 is a non-volatile memory device which typically has a small memory capacity relative to the larger memory capacity of secondary storage 604. The RAM 608 is used to store volatile data and perhaps to store instructions. Access to both ROM 606 and RAM 608 is typically faster than to secondary storage 604. The secondary storage 604, the RAM 608, and/or the ROM 606 may be referred to in some contexts as computer readable storage media and/or non-transitory computer readable media.

[0043] I/O devices 610 may include printers, video monitors, liquid crystal displays (LCDs), touch screen displays, keyboards, keypads, switches, dials, mice, track balls, voice recognizers, card readers, paper tape readers, or other well-known input devices.

[0044] The network connectivity devices 612 may take the form of modems, modem banks, Ethernet cards, universal serial bus (USB) interface cards, serial interfaces, token ring cards, fiber distributed data interface (FDDI) cards, wireless local area network (WLAN) cards, radio transceiver cards such as code division multiple access (CDMA), global system for mobile communications (GSM), long-term evolution (LTE), worldwide interoperability for microwave access (WiMAX), and/or other air interface protocol radio transceiver cards, and other well-known network devices. These network connectivity devices 612 may enable the processor 602 to communicate with the Internet or one or more intranets. With such a network connection, it is contemplated that the processor 602 might receive information from the network, or might output information to the network in the course of performing the above-described method steps. Such information, which is often represented as a sequence of instructions to be executed using processor 602, may be received from and outputted to the network, for example, in the form of a computer data signal embodied in a carrier wave.

[0045] Such information, which may include data or instructions to be executed using processor 602 for example, may be received from and outputted to the network, for example, in the form of a computer data baseband signal or signal embodied in a carrier wave. The baseband signal or signal embedded in the carrier wave, or other types of signals currently used or hereafter developed, may be generated according to several methods known to one skilled in the art. The

baseband signal and/or signal embedded in the carrier wave may be referred to in some contexts as a transitory signal.

[0046] The processor 602 executes instructions, codes, computer programs, scripts which it accesses from hard disk, floppy disk, optical disk (these various disk based systems may all be considered secondary storage 604), ROM 606, RAM 608, or the network connectivity devices 612. While only one processor 602 is shown, multiple processors may be present. Thus, while instructions may be discussed as executed by a processor, the instructions may be executed simultaneously, serially, or otherwise executed by one or multiple processors. Instructions, codes, computer programs, scripts, and/or data that may be accessed from the secondary storage 604, for example, hard drives, floppy disks, optical disks, and/or other device, the ROM 606, and/or the RAM 608 may be referred to in some contexts as non-transitory instructions and/or non-transitory information.

[0047] In an embodiment, the computer system 600 may comprise two or more computers in communication with each other that collaborate to perform a task. For example, but not by way of limitation, an application may be partitioned in such a way as to permit concurrent and/or parallel processing of the instructions of the application. Alternatively, the data processed by the application may be partitioned in such a way as to permit concurrent and/or parallel processing of different portions of a data set by the two or more computers. In an embodiment, virtualization software may be employed by the computer system 600 to provide the functionality of a number of servers that is not directly bound to the number of computers in the computer system 600. For example, virtualization software may provide twenty virtual servers on four physical computers. In an embodiment, the functionality disclosed above may be provided by executing the application and/or applications in a cloud computing environment. Cloud computing may comprise providing computing services via a network connection using dynamically scalable computing resources. Cloud computing may be supported, at least in part, by virtualization software. A cloud computing environment may be established by an enterprise and/or may be hired on an as-needed basis from a third party provider. Some cloud computing environments may comprise cloud

computing resources owned and operated by the enterprise as well as cloud computing resources hired and/or leased from a third party provider.

[0048] In an embodiment, some or all of the reflective memory bridge functionality disclosed above may be provided as a computer program product. The computer program product may comprise one or more computer readable storage medium having computer usable program code embodied therein to implement the functionality disclosed above. The computer program product may comprise data structures, executable instructions, and other computer usable program code. The computer program product may be embodied in removable computer storage media and/or non-removable computer storage media. The removable computer readable storage medium may comprise, without limitation, a paper tape, a magnetic tape, magnetic disk, an optical disk, a solid state memory chip, for example analog magnetic tape, compact disk read only memory (CD-ROM) disks, floppy disks, jump drives, digital cards, multimedia cards, and others. The computer program product may be suitable for loading, by the computer system 600, at least portions of the contents of the computer program product to the secondary storage 604, to the ROM 606, to the RAM 608, and/or to other non-volatile memory and volatile memory of the computer system 600. The processor 602 may process the executable instructions and/or data structures in part by directly accessing the computer program product, for example by reading from a CD-ROM disk inserted into a disk drive peripheral of the computer system 600. Alternatively, the processor 602 may process the executable instructions and/or data structures by remotely accessing the computer program product, for example by downloading the executable instructions and/or data structures from a remote server through the network connectivity devices 612. The computer program product may comprise instructions that promote the loading and/or copying of data, data structures, files, and/or executable instructions to the secondary storage 604, to the ROM 606, to the RAM 608, and/or to other non-volatile memory and volatile memory of the computer system 600.

[0049] In some contexts, the secondary storage 604, the ROM 606, and the RAM 608 may be referred to as a non-transitory computer readable medium or a

computer readable storage media. A dynamic RAM embodiment of the RAM 608, likewise, may be referred to as a non-transitory computer readable medium in that while the dynamic RAM receives electrical power and is operated in accordance with its design, for example during a period of time during which the computer 600 is turned on and operational, the dynamic RAM stores information that is written to it. Similarly, the processor 602 may comprise an internal RAM, an internal ROM, a cache memory, and/or other internal non-transitory storage blocks, sections, or components that may be referred to in some contexts as non-transitory computer readable media or computer readable storage media.

[0050] In some examples, a non-transitory computer-readable storage medium may store reflective memory bridge instructions 609 that, when executed, cause the processor 602 to assign an incoming region of a local memory to at least one external computing node. The reflective memory bridge instructions 609, when executed, also may cause the processor 602 to assign an outgoing region of the local memory to at least one external computing node. The reflective memory bridge instructions 609, when executed, also may cause the processor 602 to perform reflective memory operations using the incoming region and the outgoing region. In some examples, the reflective memory bridge instructions 609, when executed, may cause the processor 602 to store for an outgoing region: a name, physical addresses associated with the outgoing region, and a set of external computing nodes that are to receive updates to the outgoing region. Further, the reflective memory bridge instructions 609, when executed, may cause the processor 602 to store for an incoming region: a name, physical addresses associating with incoming region, and a set of external computing nodes that are to reflect to the incoming region. Further, the reflective memory bridge instructions 609, when executed, may cause the processor 602 to determine a number of writes from an outgoing region that are still in a buffer by incrementing a count when a write from the outgoing region is observed and decrementing the count when a write is released from the buffer. Further, the reflective memory bridge instructions 609, when executed, may cause the processor 602 to perform other reflective memory bridge operations as described herein.

[0051] Figure 7 shows an example of a method 700 in accordance with the disclosure. The method 700 may be performed by a reflective memory bridge and/or other components of a computing node such as of the computing node 101, components of the computing node 201A-201D, or components of the external computing nodes 112 described herein. As shown, the method 700 comprises assigning an incoming region of a local memory to at least one external computing node (block 702). At block 704, the method 700 also comprises assigning an outgoing region of the local memory to at least one external computing node. At block 706, the method 700 comprises performing reflective memory operations using the incoming region and the outgoing region.

[0052] In some examples, the method 700 may comprise additional or alternative steps. For example, the method 700 may further comprise storing for an outgoing region: a name, physical addresses associated with the outgoing region, and a set of external computing nodes that are to receive updates to the outgoing region. Further, the method 700 also may comprise storing for an incoming region: a name, physical addresses associating with incoming region, and a set of external computing nodes that are to reflect to the incoming region. Further, the method 700 also may comprise determining unavailability of an interconnect or a memory region used for reflective memory operations. Further, method 700 also may comprise performing other reflective memory bridge operations as described herein.

[0053] The above discussion is meant to be illustrative of the principles and various examples of the present invention. Numerous variations and modifications will become apparent to those skilled in the art once the above disclosure is fully appreciated. It is intended that the following claims be interpreted to embrace all such variations and modifications.

CLAIMS

What is claimed is:

1. A computing node, comprising:
a processor;
local memory coupled to the processor;
a reflective memory bridge coupled to the processor;
wherein the reflective memory bridge maps to an incoming region of the local memory assigned to at least one external computing node and maps to an outgoing region of the local memory assigned to at least one external computing node.
2. The computing node of claim 1, wherein the reflective memory bridge is part of a coherency fabric for the computing node and is integrated with a component selected from a list consisting of the processor, a node controller for the computing node, a coherency bus for the computing node, and personal component interconnect (PCI) card for the computing node.
3. The computing node of claim 1, wherein the reflective memory bridge specifies for the outgoing region: a name, addresses associated with the outgoing region, and a set of external computing nodes that are to receive updates to the outgoing region.
4. The computing node of claim 1, wherein the reflective memory bridge specifies for the incoming region: a name, addresses associated with the incoming region, and a set of external computing nodes that are to reflect to the incoming region.
5. The computing node of claim 1, wherein stores from the reflective memory bridge are not reflected.

6. The computing node of claim 1, further comprising:
a buffer;
a memory mapped register; and
an interconnect controller,
wherein the memory mapped register is to count a number of writes from
the outgoing region that are still in the buffer,
wherein the count is incremented when a write from the outgoing region is
observed and is decremented when a write is released from the
buffer,
wherein the interconnect controller receives a notification if fill level for the
buffer is more than a threshold amount.
7. The computing node of claim 1, further comprising a memory mapped
interface that allows remote read/exchange constructs or remote read/increment
constructs, and that transmits a timeout or error notification to an interconnect
controller when the outgoing region or the incoming region is determined to be
unavailable.
8. The computing node of claim 1, further comprising a database that utilizes
the reflective memory bridge to perform at least one operation selected from the
list consisting of:
communicating commands or results to at least one external computing
node;
distributing logs, updates, and new data to at least one external computing
node;
communicating information for two-phase commits to at least one other
computing node;
maintaining k-safety by reflecting at least part of a projection being written
to a mirror node; and
writing to a logging appliance.

9. A method for a computing node, comprising:
assigning, by the computing node, an incoming region of a local memory to at least one external computing node;
assigning, by the computing node, an outgoing region of the local memory to at least one external computing node; and
performing, by the computing node, reflective memory operations using the incoming region and the outgoing region.
10. The method of claim 9, further comprising storing for the outgoing region: a name, addresses associated with the outgoing region, and a set of external computing nodes that are to receive updates to the outgoing region.
11. The method of claim 9, further comprising storing for the incoming region: a name, addresses associating with incoming region, and a set of external computing nodes that are to reflect to the incoming region.
12. The method of claim 9, further comprising determining unavailability of an interconnect or a memory region used for reflective memory operations.
13. An integrated circuit, comprising:
a reflective memory bridge to manage communications to an incoming region of a local memory of a computing node assigned to at least one external computing node and to manage communications to an outgoing region of the local memory assigned to at least one external computing node.
14. The integrated circuit of claim 13, wherein the integrated circuit is part of a processor for the computing node.
15. The integrated circuit of claim 13, wherein the integrated circuit is part of a node controller for the computing node.

1/10

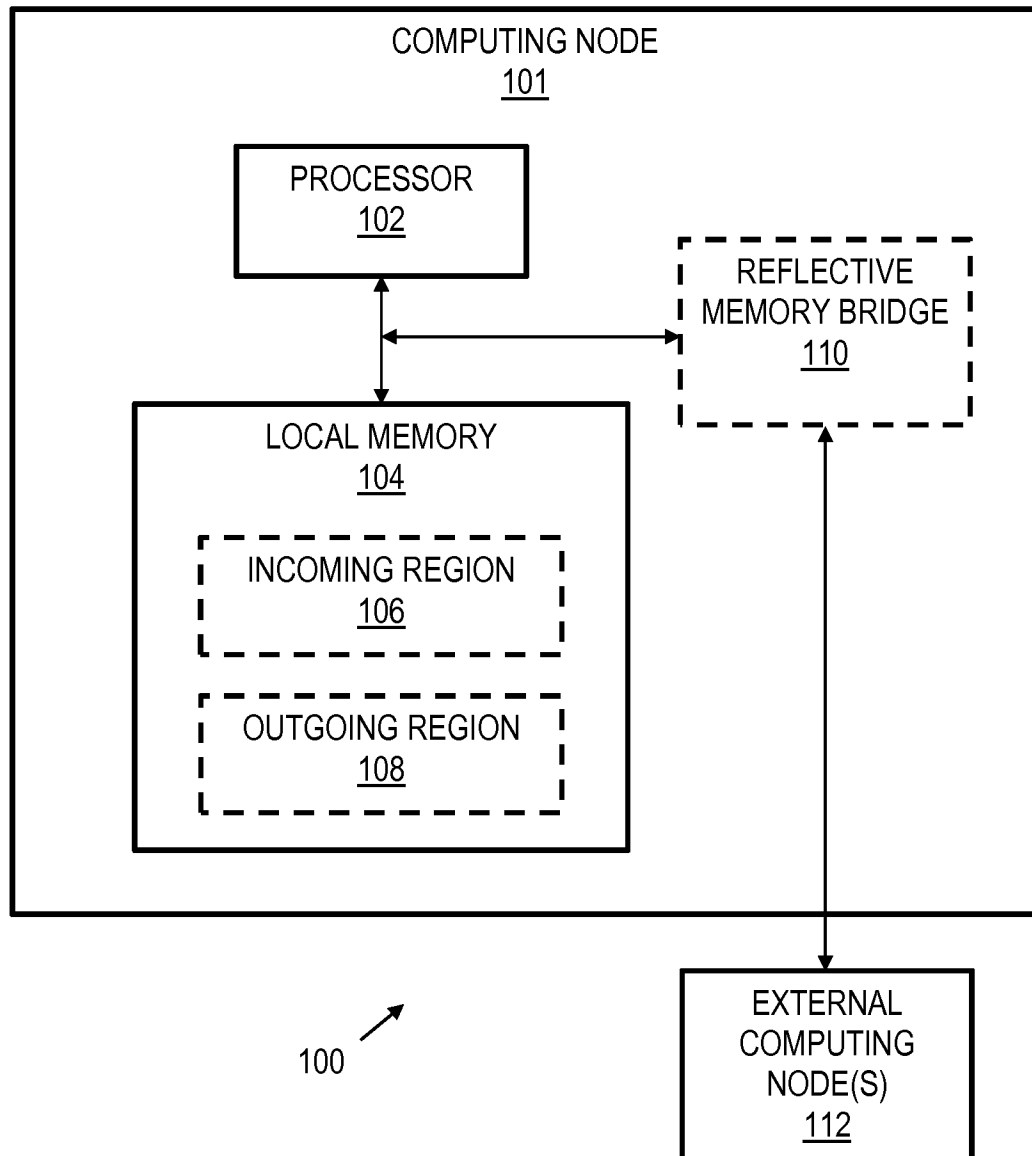
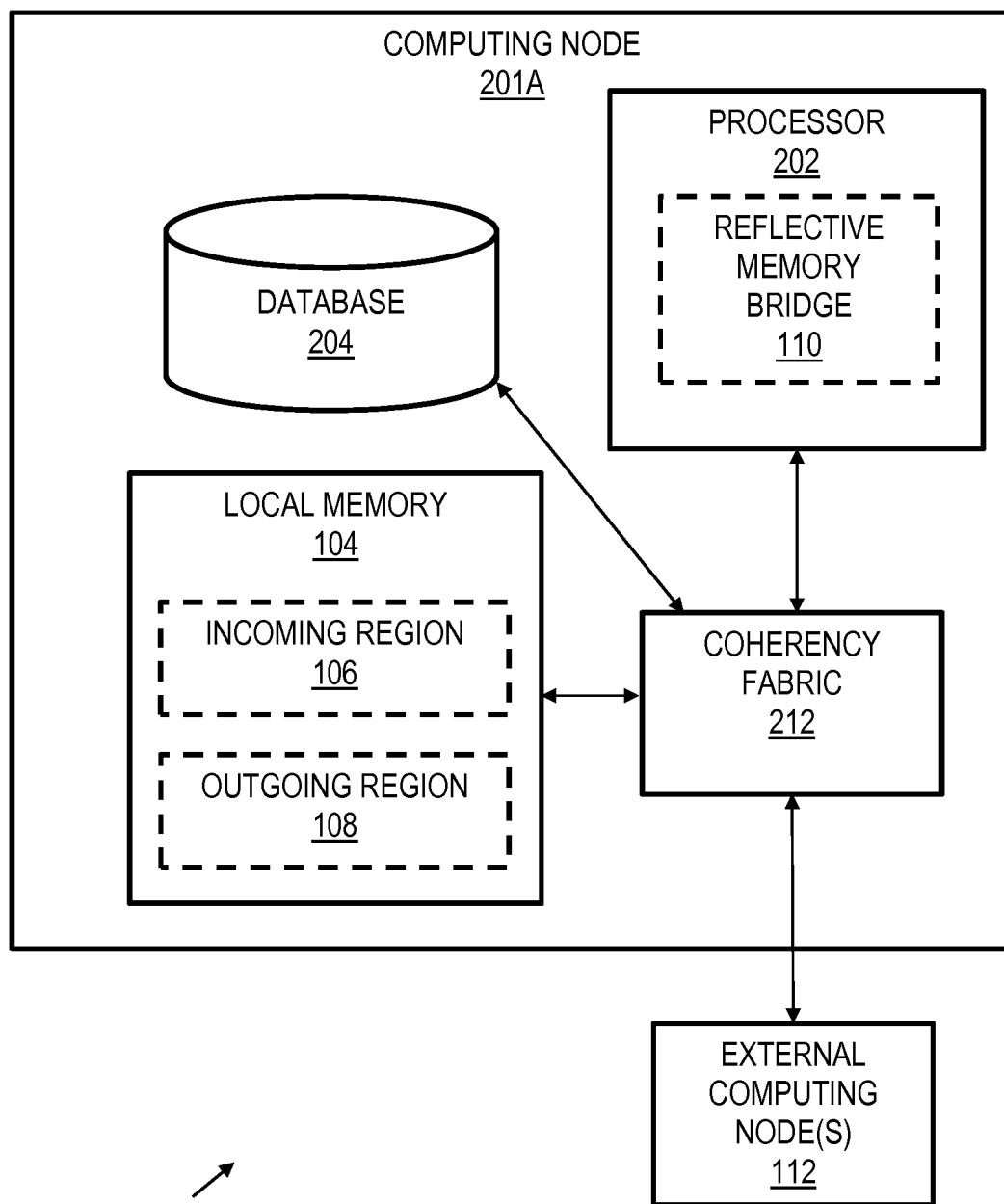


FIG. 1

2/10



200A ↗

FIG. 2A

3/10

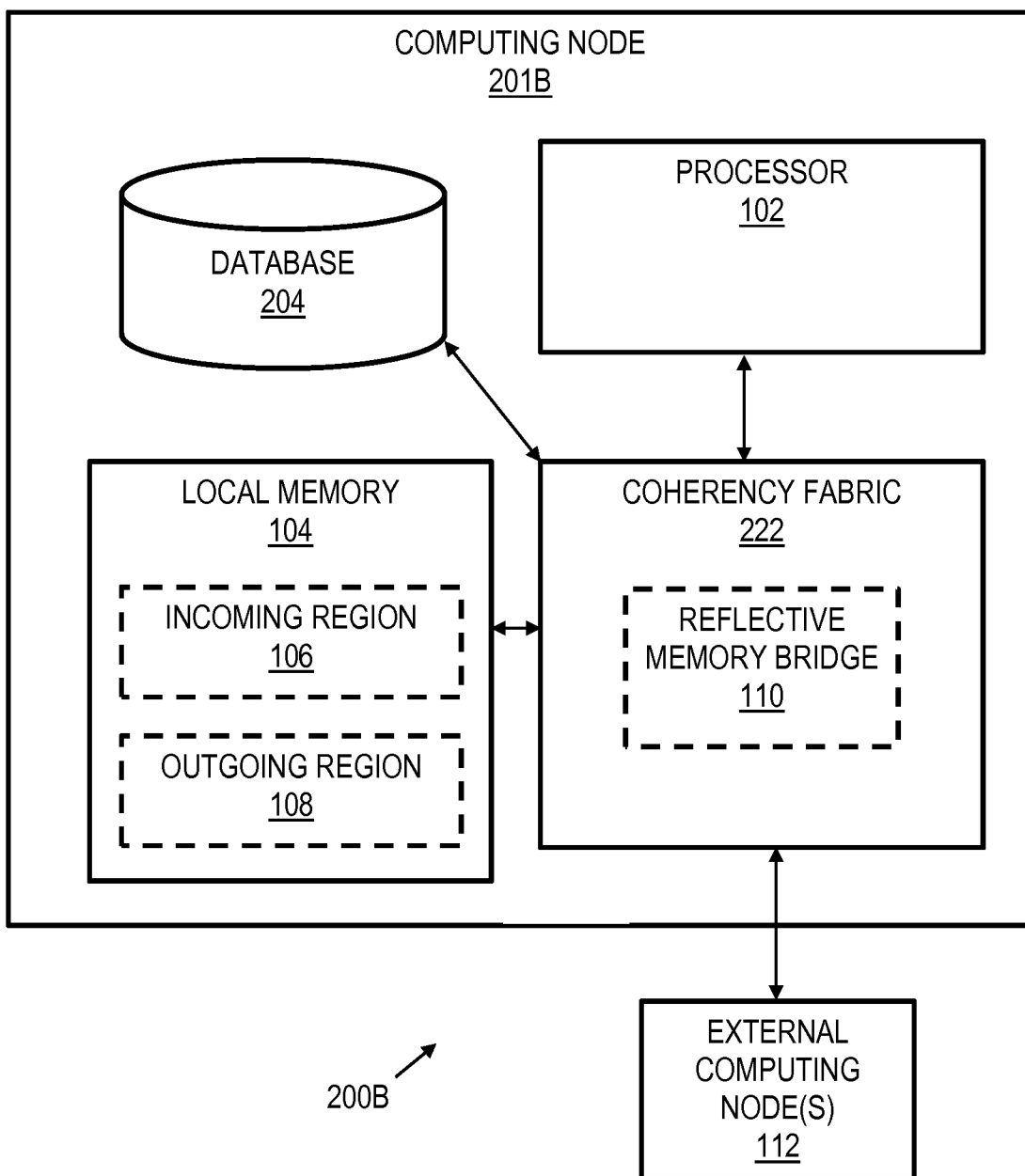


FIG. 2B

4/10

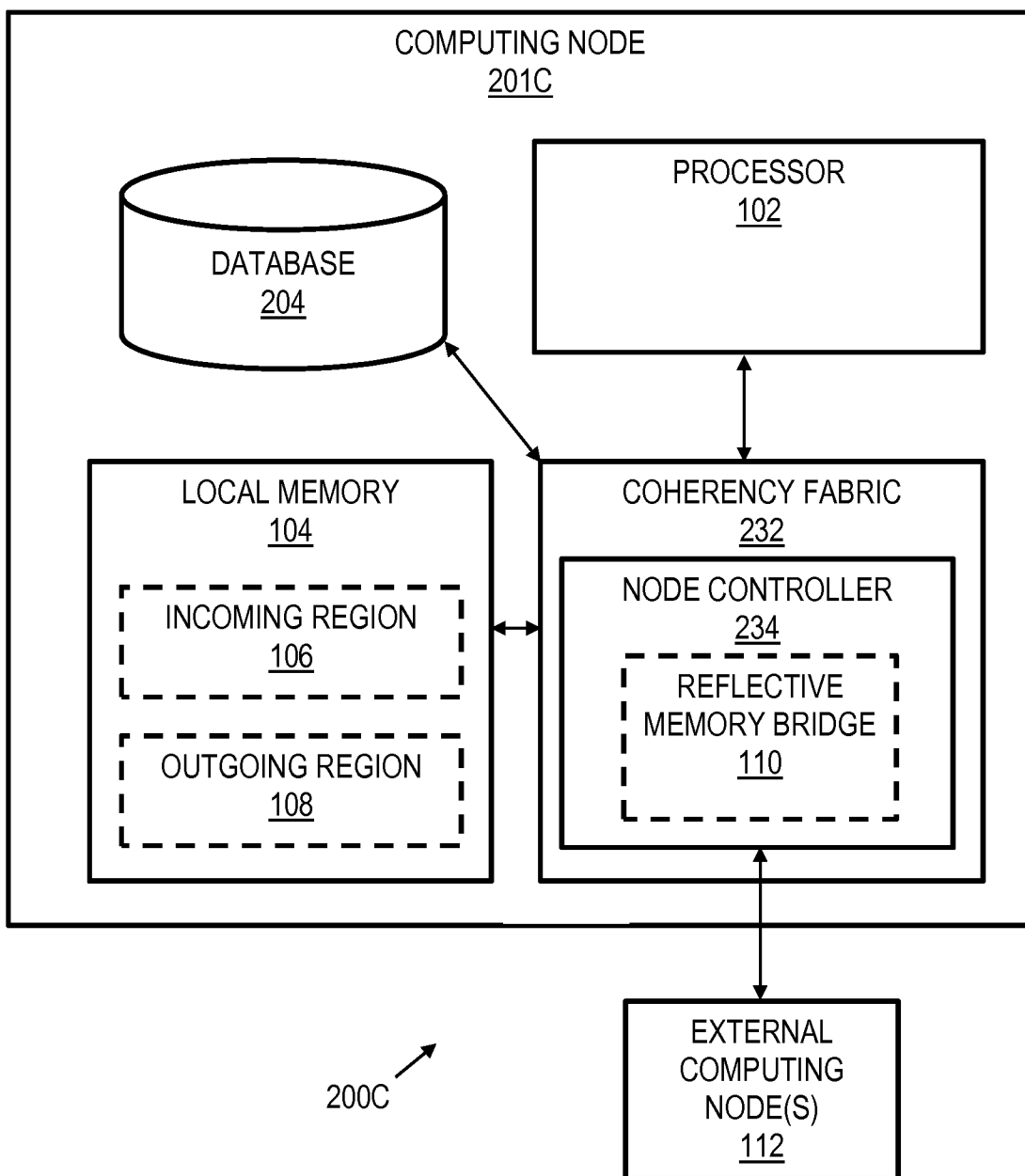


FIG. 2C

5/10

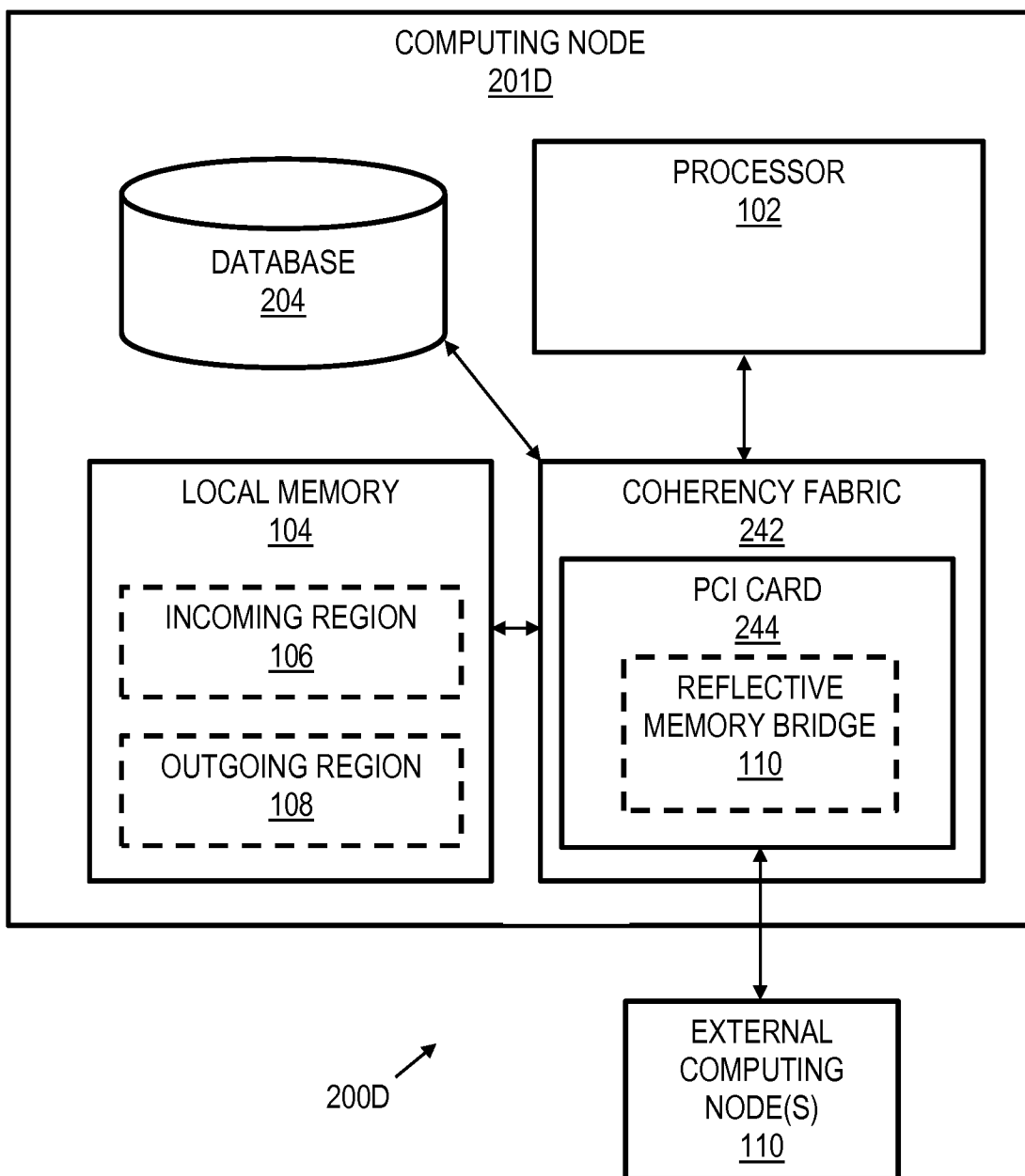


FIG. 2D

6/10

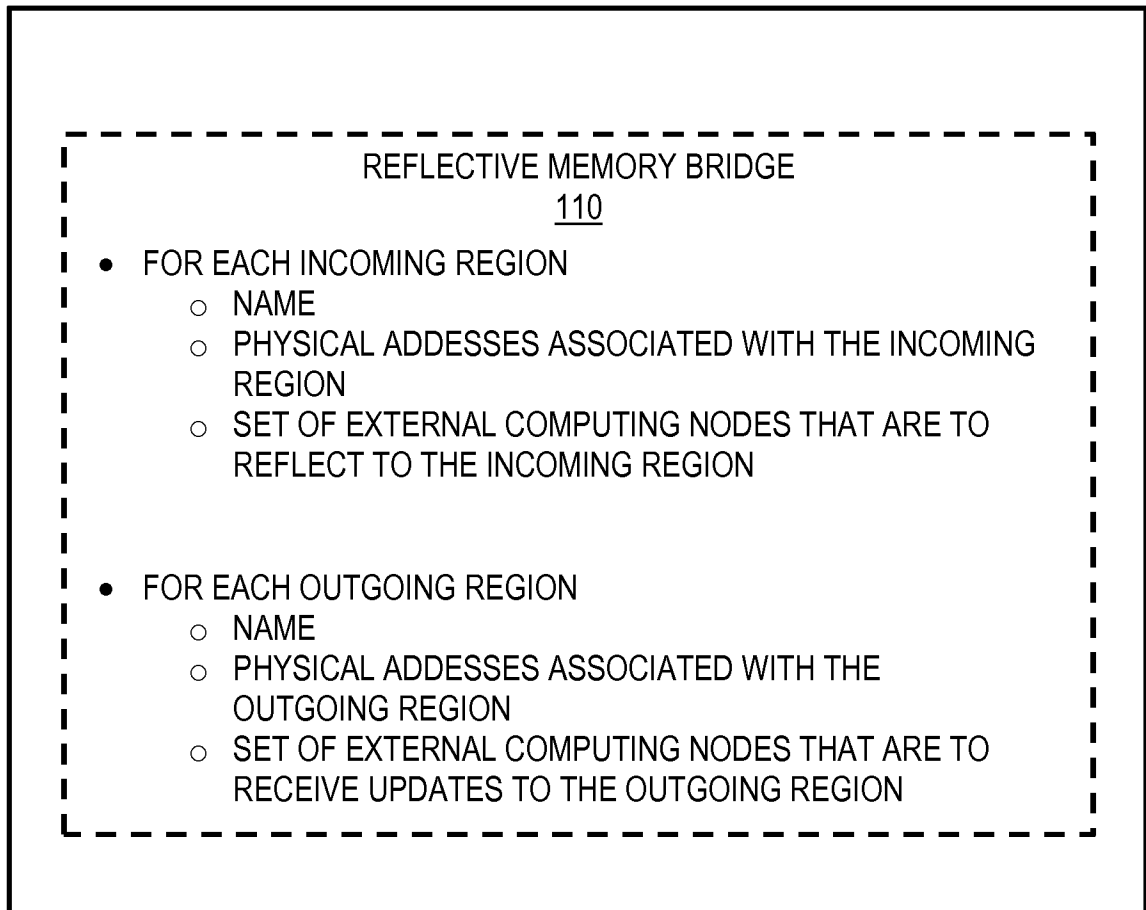


FIG. 3

7/10

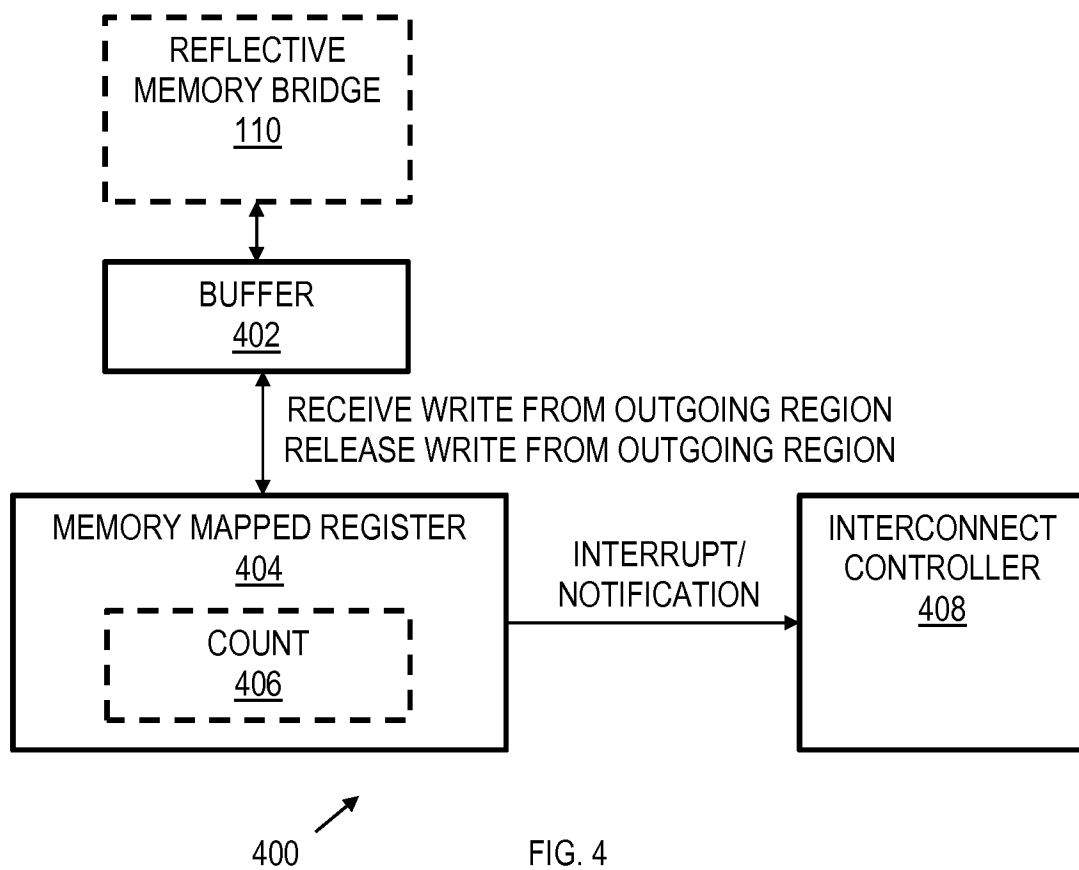


FIG. 4

8/10

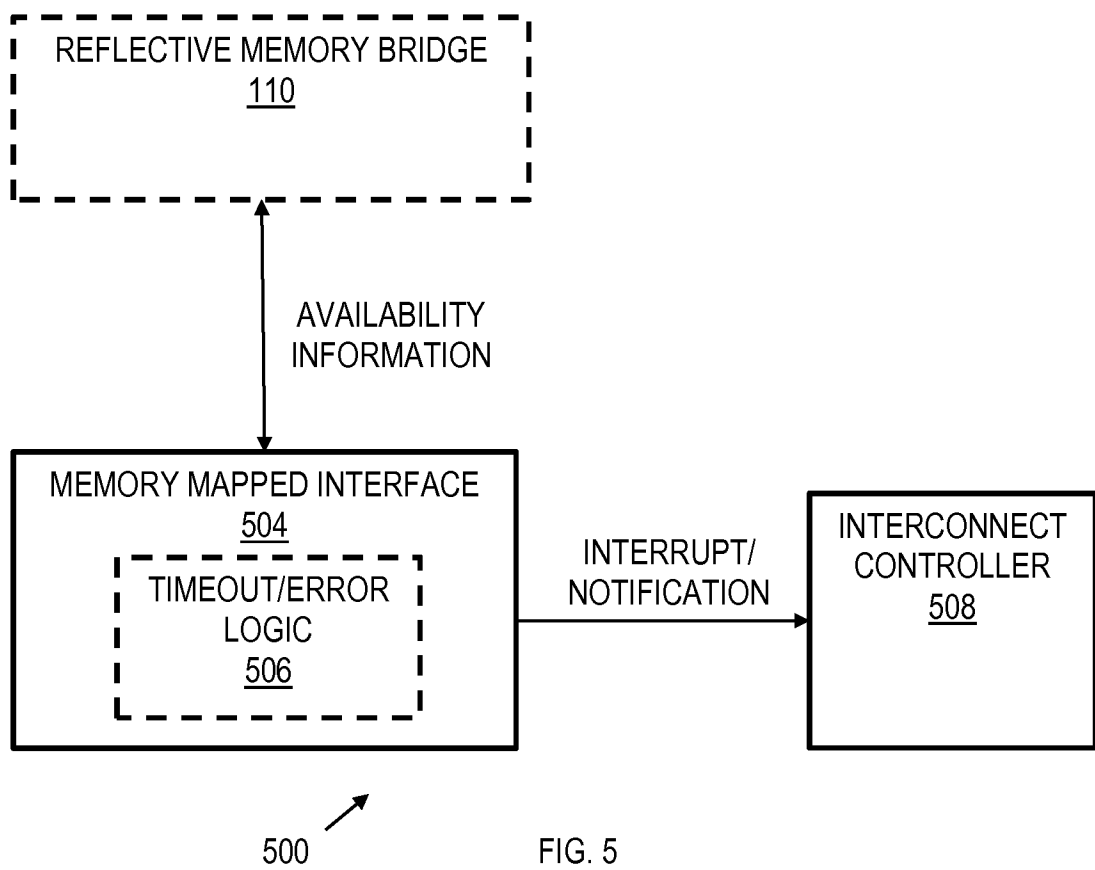
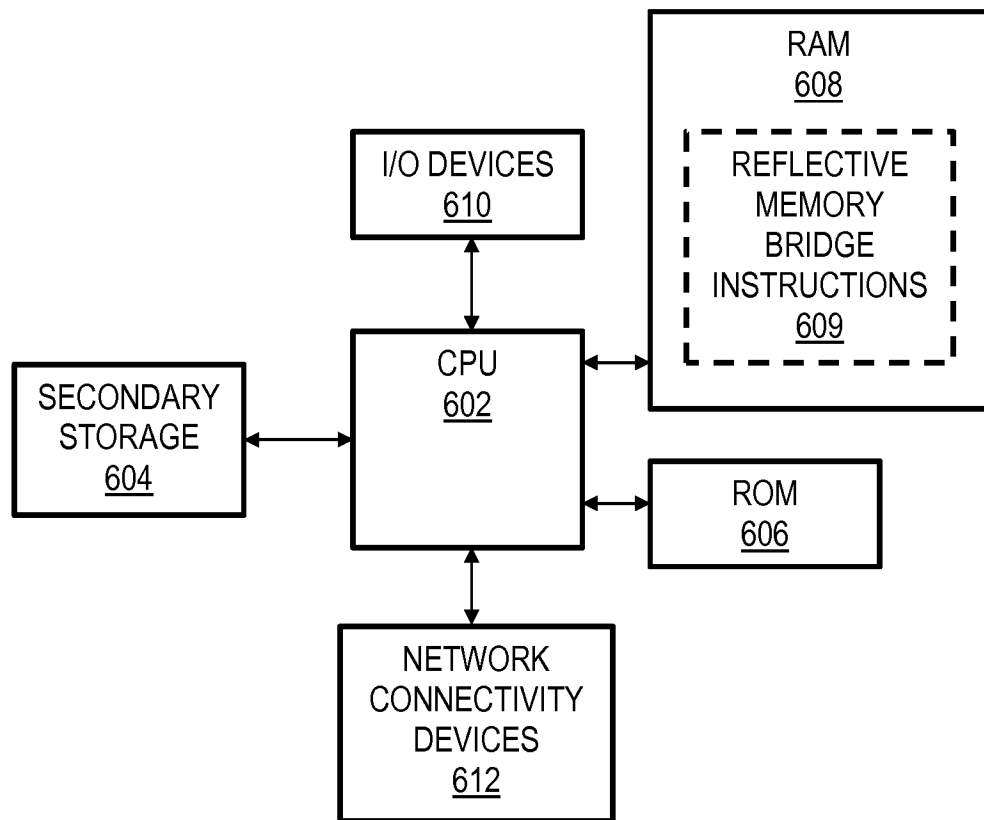


FIG. 5

9/10



600 ↗

FIG. 6

10/10

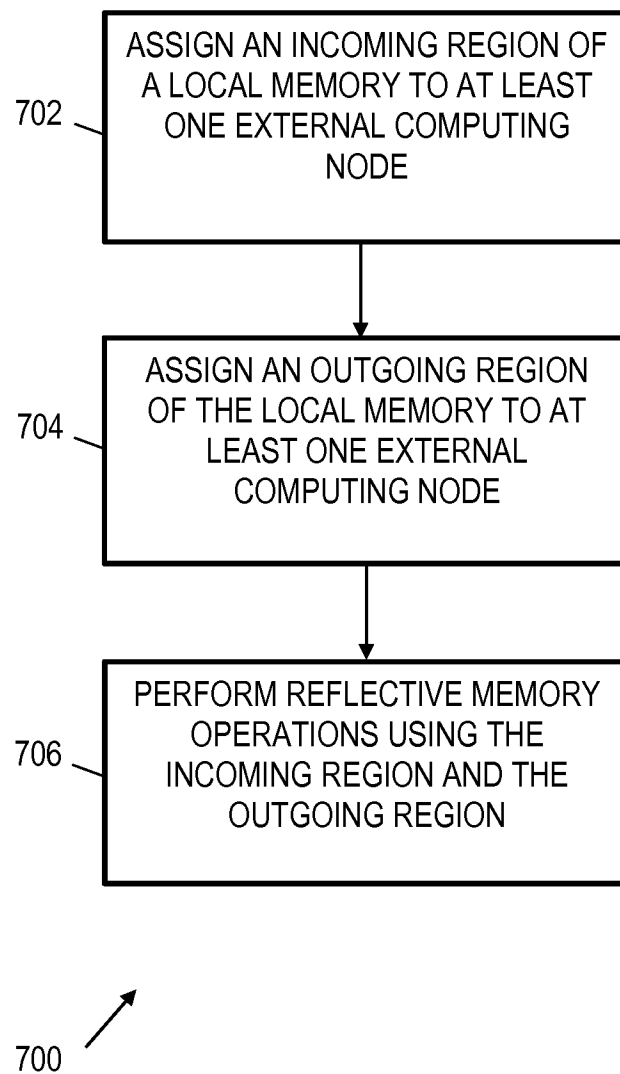


FIG. 7

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2012/035904**A. CLASSIFICATION OF SUBJECT MATTER****G06F 13/14(2006.01)i, G06F 13/38(2006.01)i, G06F 15/16(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 13/14; G06F 12/16; G06B 9/02; G06F 19/00; G06F 12/00; G06F 15/167

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:reflect*,memory*,stor*,processor*,map*,coupl*,connect*,data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 05581732A A (DANN; JAMES C.) 03 December 1996 See Column 4, lines 38-46 and Claim 1.	1-15
A	US 06067595A A (LINDENSTRUTH; VOLKER) 23 May 2000 See Column 8, lines 6-12 and Column 8, lines 46-51.	1-15
A	US 2003-0110233 A1 (SCOTT D. PRALL et al.) 12 June 2003 See Paragraphs [0008]-[0009] and claims 1-23.	1-15
A	US 05984504A A (DOYLE; JAMES P. et al.) 16 November 1999 See Claims 1-23.	1-15



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of 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

20 DECEMBER 2012 (20.12.2012)

Date of mailing of the international search report

21 DECEMBER 2012 (21.12.2012)

Name and mailing address of the ISA/KR

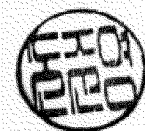
Korean Intellectual Property Office
189 Cheongsu-ro, Seo-gu, Daejeon Metropolitan
City, 302-701, Republic of Korea

Facsimile No. 82-42-472-7140

Authorized officer

Son, June Young

Telephone No. 82-42-481-5720



INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2012/035904

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 05581732A A	03.12.1996	JP 05-070182 B	04.10.1993
		JP 1861874 C	08.08.1994
		JP 61-000856 A	06.01.1986
		US 04991079A A	05.02.1991
		US 05072373A A	10.12.1991
		US 05255369A A	19.10.1993
US 06067595A A	23.05.2000	AU 1998-96657 A1	12.04.1999
		CN 1154934 C0	23.06.2004
		CN 1279785 A0	10.01.2001
		EP 1047994 A2	02.11.2000
		EP 1047994 A4	24.07.2002
		EP 1047994 B1	16.04.2008
		JP 2001-517844 A	09.10.2001
		KR 10-2001-0015608 A	26.02.2001
		WO 99-16200 A2	01.04.1999
US 2003-0110233 A1	12.06.2003	None	01.04.1999
		None	
US 05984504A A	16.11.1999	None	