



- (51) **International Patent Classification:**
G06F 13/14 (2006.01) **G06F 12/00** (2006.01)
G06F 13/16 (2006.01)
- (21) **International Application Number:**
PCT/US2013/048079
- (22) **International Filing Date:**
27 June 2013 (27.06.2013)
- (25) **Filing Language:** English
- (26) **Publication Language:** English
- (30) **Priority Data:**
13/716,801 17 December 2012 (17.12.2012) US
- (71) **Applicant:** INTEL CORPORATION [US/US]; 2200 Mission College Boulevard, Santa Clara, California 95054 (US).
- (72) **Inventors:** VARMA, Ankush; 179 SE 49th Court, Hillsboro, Oregon 97123 (US). SISTLA, Krishnakanth V.; 6000 SW Fountain Grove Terrace, Beaverton, Oregon 97007 (US).
- (74) **Agent:** ROZMAN, Mark J.; Trop, Pruner & Hu, P.C., 1616 S. Voss Rd., Ste. 750, Houston, Texas 77057-2631 (US).
- (81) **Designated States** (*unless otherwise indicated, for every kind of national protection available*): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, 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, PA, 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:** PERFORMING FREQUENCY COORDINATION IN A MULTIPROCESSOR SYSTEM BASED ON RESPONSE TIMING OPTIMIZATION

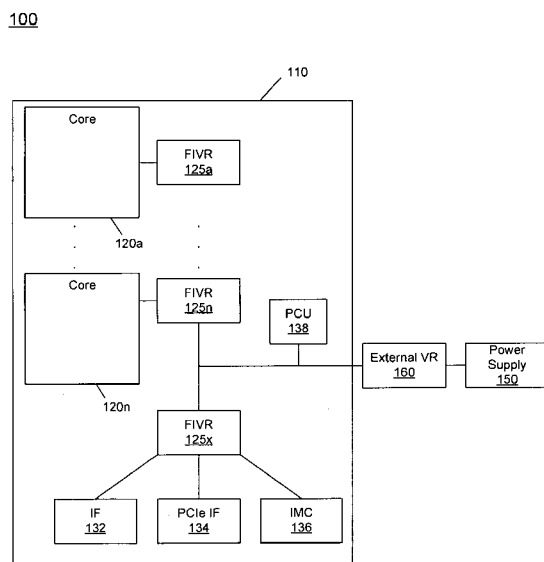


FIG. 1

(57) **Abstract:** In an embodiment, a processor includes a core to execute instructions and a logic to receive memory access requests from the core and to route the memory access requests to a local memory and to route snoop requests corresponding to the memory access requests to a remote processor. The logic is configured to maintain latency information regarding a difference between receipt of responses to the snoop requests from the remote processor and receipt of responses to the memory access requests from the local memory. Other embodiments are described and claimed.



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

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

Declarations under Rule 4.17:

- *as to the identity of the inventor (Rule 4.17(i))*
- *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

Published:

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

PERFORMING FREQUENCY COORDINATION IN A
MULTIPROCESSOR SYSTEM BASED ON RESPONSE TIMING OPTIMIZATION

Background

[0001] Advances in semiconductor processing and logic design have permitted an increase in the amount of logic that may be present on integrated circuit devices. As a result, computer system configurations have evolved from a single or multiple integrated circuits in a system to multiple hardware threads, multiple cores, multiple devices, and/or complete systems on individual integrated circuits. Additionally, as the density of integrated circuits has grown, the power requirements for computing systems (from embedded systems to servers) have also escalated. Furthermore, software inefficiencies, and its requirements of hardware, have also caused an increase in computing device energy consumption. In fact, some studies indicate that computing devices consume a sizeable percentage of the entire electricity supply for a country, such as the United States of America. As a result, there is a vital need for energy efficiency and conservation associated with integrated circuits. These needs will increase as servers, desktop computers, notebooks, Ultrabooks™, tablets, mobile phones, processors, embedded systems, etc. become even more prevalent (from inclusion in the typical computer, automobiles, and televisions to biotechnology).

[0002] In a multiprocessor system, latency of an access to a remote processor is determined by the remote processor's frequency, more specifically, a frequency of a non-core portion of the processor, often referred to as an uncore. Standard power management features reduce uncore frequency when cores are inactive. This creates a scenario such that when cores of one processor seek to access another processor and the remote processor has idle/inactive cores (and thus low uncore frequency), this low remote frequency will cause an increase in the response time. While the remote processor can maintain a fixed frequency for this uncore frequency, this can increase power consumption and still may not resolve the problem of high latency, particularly in low bandwidth workloads.

Brief Description of the Drawings

[0003] FIG. 1 is a block diagram of a portion of a system in accordance with an embodiment of the present invention.

[0004] FIG. 2 is a flow diagram of a method in accordance with an embodiment of the present invention.

[0005] FIG. 3A is a flow diagram of a method for requesting frequency coordination in accordance with an embodiment of the present invention.

[0006] FIG. 3B is a flow diagram of a method for performing an uncore frequency update in accordance with an embodiment of the present invention.

[0007] FIG. 4A is a block diagram of an example two processor system in accordance with an embodiment of the present invention.

[0008] FIG. 4B is a block diagram illustrating a flow of communications regarding handling of a memory access request in accordance with an embodiment of the present invention.

[0009] FIG. 5 is a block diagram of a processor in accordance with an embodiment of the present invention.

[0010] FIG. 6 is a block diagram of a processor in accordance with an embodiment of the present invention.

[0011] FIG. 7 is a block diagram of a system in accordance with an embodiment of the present invention.

Detailed Description

[0012] In various embodiments, a detection mechanism can detect a condition in which response latency for a response received in a first processor is constrained by a remote processor. Responsive to this detection a communication can be generated and sent to the remote processor. In turn, operations may occur at the remote processor to increase uncore frequency to alleviate the performance loss. These processors may be respective processor sockets of a multiprocessor system

such as a given personal computer, server computer, mobile device, or other computing device.

[0013] In an embodiment a sensor may count transactions where performance (or response latency) is sensitive to a frequency of the remote processor. That is, a processor includes a home agent configured to receive, responsive to a memory access transaction, a snoop response from the remote processor and data from an access to local memory. The home agent will frequently have to wait for both of these responses to be received to check that there are no conflicts prior to continuing with the transaction. Only the latency of the snoop response from the remote processor is determined by the remote processor frequency.

[0014] A set of counters may be provided to aid in quantifying the performance benefit possible from speeding up a frequency of the remote processor. In an embodiment, a first counter is configured to count the number of transactions where a remote snoop response was received after a local memory response. And a second counter is configured to count the total number of cycles where the home agent (or other coherency controller) waited for the remote snoop response.

[0015] In turn, these counters can be read by a power controller of a processor such as a power control unit (PCU) and compared against programmed thresholds to determine based on this latency information if there is sufficient performance benefit to trigger an inter-processor frequency coordination request. Note that as described herein, this frequency coordination may take different forms, including a situation in which the frequency of non-core circuitry, also referred to herein as system agent or so-called uncore circuitry, is set at a common frequency across multiple processor sockets. In other situations, this frequency coordination may take the form of a request from one processor socket to another socket (or sockets) to seek an update to the uncore frequency, to improve latency of inter-processor communication. Furthermore, understand that while implementations described herein are with regard to processor sockets that can operate at independent frequencies for different domains including an uncore domain, embodiments apply equally to processors that operate at a single frequency. Furthermore, while described herein as performing frequency coordination with regard to uncore

frequency, in some embodiments this frequency coordination may be for additional or different components of a processor socket, such as an interconnect frequency in an implementation in which an interconnect frequency such as a ring interconnect of a processor socket operates at a different frequency than an uncore circuit.

[0016] Although the scope of the present invention is not limited in this regard, uncore frequency coordination control as described herein can be implemented via firmware, hardware, software, and combinations thereof, to thus enable detection of an undesired latency, communication of a request to at least one other processor socket to enable an increased uncore frequency to reduce the latency.

[0017] In an embodiment, a frequency coordination request may be a PCU-to-PCU message tunneled over an inter-processor link. When a processor receives this indication from another processor, it attempts to raise its uncore frequency, if possible within current power constraints of the processor. It may raise uncore frequency incrementally (one step at a time) or in larger steps that are based on the counter information, with higher counts triggering larger upticks in remote uncore frequency.

[0018] Note that in many embodiments, it is only beneficial to increase a remote frequency so that a remote snoop response is timed to arrive at the home agent substantially at the same time as the local memory access. After this point, the local memory becomes the latency determiner and there is no latency benefit from further increasing in remote processor frequency.

[0019] Referring now to FIG. 1, shown is a block diagram of a portion of a system in accordance with an embodiment of the present invention. As shown in FIG. 1, system 100 may include various components, including a processor 110 which as shown is a multicore processor. Processor 110 may be coupled to a power supply 150 via an external voltage regulator 160, which may perform a first voltage conversion to provide a primary regulated voltage to processor 110.

[0020] As seen, processor 110 may be a single die processor socket including multiple cores 120a – 120n. In addition, each core may be associated with an individual voltage regulator 125a – 125n. Accordingly, a fully integrated voltage

regulator (FIVR) implementation may be provided to allow for fine-grained control of voltage and thus power and performance of each individual core. As such, each core can operate at an independent voltage and frequency, enabling great flexibility and affording wide opportunities for balancing power consumption with performance.

[0021] Still referring to FIG. 1, additional components may be present within the processor including an input/output interface 132, another interface 134, and an integrated memory controller 136. As seen, each of these components may be powered by another integrated voltage regulator 125x. In one embodiment, interface 132 may be in accordance with the Intel® Quick Path Interconnect (QPI) protocol, which provides for point-to-point (PtP) links in a cache coherent protocol that includes multiple layers including a physical layer, a link layer and a protocol layer. As will be described herein via one or more QPI interfaces, uncore frequency coordination messages may be communicated between processor sockets of a multiprocessor system. In turn, interface 134 may be in accordance with a Peripheral Component Interconnect Express (PCIe™) specification, e.g., the PCI Express™ Specification Base Specification version 2.0 (published January 17, 2007).

[0022] Also shown is a power control unit (PCU) 138, which may include hardware, software and/or firmware to perform power management operations with regard to processor 110. In various embodiments, PCU 138 may include logic to perform uncore frequency control in accordance with an embodiment of the present invention. Furthermore, PCU 138 may be coupled via a dedicated interface to external voltage regulator 160. In this way, PCU 138 can instruct the voltage regulator to provide a requested regulated voltage to the processor.

[0023] While not shown for ease of illustration, understand that additional components may be present within processor 110 such as additional uncore logic and other components such as internal memories, e.g., one or more levels of a cache memory hierarchy, a coherency controller such as a home agent and so forth. Furthermore, while shown in the implementation of FIG. 1 with an integrated voltage regulator, embodiments are not so limited.

[0024] Although the following embodiments are described with reference to energy conservation and energy efficiency in specific integrated circuits, such as in computing platforms or processors, other embodiments are applicable to other types of integrated circuits and logic devices. Similar techniques and teachings of embodiments described herein may be applied to other types of circuits or semiconductor devices that may also benefit from better energy efficiency and energy conservation. For example, the disclosed embodiments are not limited to any particular type of computer systems, and may be also used in other devices, such as handheld devices, systems on chip (SoCs), and embedded applications. Some examples of handheld devices include cellular phones, Internet protocol devices, digital cameras, personal digital assistants (PDAs), and handheld PCs. Embedded applications typically include a microcontroller, a digital signal processor (DSP), network computers (NetPC), set-top boxes, network hubs, wide area network (WAN) switches, or any other system that can perform the functions and operations taught below. Moreover, the apparatus', methods, and systems described herein are not limited to physical computing devices, but may also relate to software optimizations for energy conservation and efficiency. As will become readily apparent in the description below, the embodiments of methods, apparatus', and systems described herein (whether in reference to hardware, firmware, software, or a combination thereof) are vital to a 'green technology' future, such as for power conservation and energy efficiency in products that encompass a large portion of the US economy.

[0025] Note that the remote processor latency detection and uncore frequency control described herein may be independent of and complementary to an operating system (OS)-based mechanism, such as the Advanced Configuration and Platform Interface (ACPI) standard (e.g., Rev. 3.0b, published October 10, 2006). According to ACPI, a processor can operate at various performance states or levels, namely from P0 to PN. In general, the P1 performance state may correspond to the highest guaranteed performance state that can be requested by an OS. In addition to this P1 state, the OS can further request a higher performance state, namely a P0 state. This P0 state may thus be an opportunistic state in which, when power and/or thermal budget is available, processor hardware can configure the processor or at

least portions thereof to operate at a higher than guaranteed frequency. In many implementations a processor can include multiple so-called bin frequencies above a guaranteed maximum frequency, also referred to as a P1 frequency. In addition, according to ACPI, a processor can operate at various power states or levels. With regard to power states, ACPI specifies different power consumption states, generally referred to as C-states, C0, C1 to Cn states. When a core is active, it runs at a C0 state, and when the core is idle it may be placed in a core low power state, also called a core non-zero C-state (e.g., C1-C6 states), with each C-state being at a lower power consumption level (such that C6 is a deeper low power state than C1, and so forth). Note that as more cores of a processor socket enter into deeper low power states, the activity level of the processor reduces and accordingly, so too may the uncore frequency, according to certain power management features of the processor such as implemented via the PCU.

[0026] In an embodiment, a dedicated communication protocol is provided for inter-processor frequency coordination. When the low remote processor frequency control feature is triggered in a processor, one or more messages according to this protocol are sent over an inter-processor interconnect (such as a QPI link or other point-to-point (PtP) interconnect that couples different processor sockets of a multiprocessor system). This message thus provides signaling that a processor has detected a condition that indicates inter-processor frequency coordination (and, optionally, the frequency it is requesting). In different embodiments, this communication protocol may be implemented in either a peer-to-peer manner (where each processor has the same behavior and sends requests to all other processors when it seeks frequency coordination), or in a master-slave manner (where a processor is designated as master and collects information from other processors, makes a decision, and sends requests to all other processors). In either case, the system behavior is the same.

[0027] Note that this requested updated frequency may be at a floor frequency that is deadlock-free. For example, setting the floor frequency to a maximum of the uncore frequencies of all the other processors would lead to a deadlock scenario where no processor could reduce its uncore frequency, thus adversely affecting

power savings. A corresponding deadlock-free example is to select a floor frequency that is less than a maximum of the uncore frequencies of all the other processors by a predetermined amount (such as 1 or 2 bin frequencies (where a bin frequency is 100 megahertz (MHz), in an embodiment)). In this way, processors may be allowed to reduce frequency as workload behavior changes.

[0028] Referring now to FIG. 2, shown is a flow diagram of a method in accordance with of an embodiment of the present invention. As shown in FIG. 2, method 200 may be performed by logic of a home agent or other circuitry of a processor. Note that these operations shown in FIG. 2 are with regard to a home agent that owns the local memory and provides responses to various memory requests from agents such as cores of the local processor socket. In general, method 200 may be used to analyze information regarding receipt of responses to various transactions in order to determine whether an excessive latency is occurring with regard to such responses received from remote processor sockets. As seen, method 200 begins by receiving a read request in a home agent (block 205). This read request may come from a core of the processor, which seeks to read a given piece of data located at a particular memory address. Assume for purposes of discussion that this memory address is located in a portion of a system memory local to the processor. In the course of performing this read request, one or more snoop requests are sent to remote processor sockets to determine whether any copies of this requested data are present and if so whether they are in a modified state.

[0029] Still referring to FIG. 2, control passes to block 210 where the memory access can be sent to the memory and a snoop request can be sent to a remote processor socket. For example, assume a two processor socket system. Thus in addition to sending the read request via a memory controller to the local memory, a snoop request can be sent from the home agent to the remote socket to thus request a status of the requested data in that processor (if present). Next it can be determined whether both requests have been received in the home agent (diamond 215).

[0030] Control next passes to diamond 220 to determine whether the remote processor response was received after the memory response. Here, the memory

response may be a read completion that includes the requested data. In turn, the response from the other processor socket is a snoop response indicating whether the requested data was present in a cache memory of the other processor socket and if so, its state.

[0031] If it is determined that the remote processor socket response is received after the memory response, control passes to block 225 where two counters can be incremented based on this delay in receiving the remote processor response. Specifically, these counters, which may be part of latency detection logic of a home agent, includes a first counter that maintains a count of transactions for which a response from a remote processor socket is delayed as compared to a response from local memory. The second counter may maintain a count of the number of cycles of these delayed responses. Thus for each response, a single increment can occur to the first counter while a variable amount of increment occurs to the second counter, namely corresponding to the difference in cycles between receipt of the memory response from the local memory and receipt of the corresponding snoop response from the remote processor socket. In either event, control passes to block 230 where the requested read data can be provided to the requester. Although shown at this high level in the embodiment of FIG. 2, understand the scope of the present invention is not limited in this regard.

[0032] Based on the count values, either this logic of the home agent or other processor logic (such as of a PCU) can determine whether the latency for receiving responses from one or more remote processor sockets is at an undesired high level and if so to take appropriate corrective action. In an embodiment, uncore frequency control logic of a PCU can read these counters at a predetermined interval, and based on the values of these counters determine whether a request for an updated uncore frequency in the one or more remote processor sockets is warranted. If so, the request can be generated and sent to the appropriate processor socket(s).

[0033] Referring now to FIG. 3A, shown is a flow diagram of a method for requesting frequency coordination in accordance with an embodiment of the present invention. Note that in FIG. 3A, method 240 can be implemented within various logic of a processor, such as PCU logic or a dedicated uncore frequency control logic,

which may be a standalone logic or can be incorporated into a PCU, in some implementations. As seen, method 240 begins by determining whether remote processor response latency is greater than a threshold (diamond 245). As described above, values stored in a set of counters can be read to determine whether the remote processor latency exceeds this latency threshold. Although described in this embodiment with this counter-based latency determination, in other embodiments different detection mechanisms are possible to detect undesired remote processor latency.

[0034] Referring still to FIG. 3A, next at block 250 a frequency update request may be generated. This request can take different forms in various embodiments. For example, a request to update an uncore frequency can be generated, e.g., with a preprogrammed value for the updated uncore frequency. Or a calculation can be made. In yet other embodiments, a simple request to update frequency can be generated that in turn enables the receiving processor to perform a calculation as to an appropriate uncore frequency.

[0035] Still referring to FIG. 3A, control passes next to block 255 where the frequency update request can be communicated to one or more processors coupled to the processor. As discussed above, this communication may be via a PtP interconnect that couples the processors such as a QPI or other interconnect. Different types of messages can be sent, including, in an embodiment, a frequency control message that includes an encoding for this requested operation as well as an identification of the requested updated frequency value. Although shown at this high level in the embodiment of FIG. 3A, understand the scope of the present invention is not limited in this regard.

[0036] Referring now to FIG. 3B, shown is a flow diagram of a method for performing an uncore frequency update in accordance with an embodiment of the present invention. This method can be performed by uncore frequency control logic of a processor, as with the above discussion of FIG. 3A, in some embodiments. However understand that this uncore frequency control logic that handles incoming requests for increased uncore frequency may be different than logic of a PCU that initiates a request to a remote processor for increased frequency within the remote

processor. As seen, method 260 begins by receiving an uncore frequency request from a remote processor (block 265). As described above, this request can be received via a PtP interconnect and may include information such that the request can be directed to be received within the uncore frequency control logic. Note that this request may include a requested updated frequency or can simply be a request to update the uncore frequency. Next, control passes to diamond 270 where it can be determined whether the processor can update its uncore frequency to the requested uncore frequency, within various constraints of the processor. This determination may be made within the uncore frequency control logic, e.g., with input from a PCU that thus provides information regarding various constraints on the processor such as power and/or thermal constraints or the PCU may indicate whether an increased uncore frequency is allowed. In this way, the uncore frequency control logic can determine whether the requested uncore frequency (when the request includes a requested frequency value) can be effected. If so, control passes to block 275 where the uncore frequency can be updated to the requested uncore frequency. Thus this updated uncore frequency is at a greater value than a uncore frequency determined by the PCU for a current level of a workload being executed on the processor. Note here that this updated uncore frequency may be at a value higher (and potentially much higher) than a frequency needed to handle current activity of this local processor. Stated another way, this increased uncore frequency is to enable greater system performance overall, although it is not needed to handle the local processor workload.

[0037] Otherwise, if the processor cannot perform the update to the requested uncore frequency, control passes instead to block 280 where an available uncore frequency can be determined according to the processor constraints. The uncore frequency control logic can perform this determination or in some embodiments the PCU itself may make this determination. Finally, control passes to block 290 where the uncore frequency can be updated to the determined available uncore frequency. Although shown at this high level in the embodiment of FIG. 3B, understand the scope of the present invention is not limited in this regard.

[0038] Referring now to FIG. 4A, shown is a block diagram of an example two processor system in accordance with an embodiment of the present invention. As shown in FIG. 4A, system 300 is a multiprocessor system including a first processor socket 310a and a second processor socket 310b. Each of the sockets may be configured the same and thus only the components of processor socket 310a are discussed, although the same components are present in socket 310b. Of course in other embodiments, the processor sockets may be heterogeneous sockets in that different types of processors may be present, such as having different numbers of cores and other processing engines, differently sized cache memories, among other differences.

[0039] As seen, processor socket 310a includes a plurality of cores 315_{ao}-315_{an} that couple to other components of the socket via an interconnect 320a which in an embodiment can be a ring interconnect. Each of the cores may be of a common architecture or there can be cores of multiple micro-architectures. Each such core may include a local cache memory. In addition via interconnect 320a the cores are coupled to a cache memory 340a which in an embodiment is a shared cache memory such as a last level cache (LLC).

[0040] Also via interconnect 320a the cores couple to a home agent 325a that acts as a coherency controller, and includes latency detection logic in accordance with an embodiment of the present invention (not shown in FIG. 4A for ease of illustration). This latency detection logic is configured to determine an amount of latency for responses from one or more remote processor sockets to requests such as snoop requests issued by the home agent. More specifically, this latency may be a duration longer than receipt of a corresponding response received from a local memory for the same transaction. Using latency detection logic of the home agent, it can be determined whether excessive latencies are occurring as to responses received from one or more processor sockets and if so, an indication can be made to an appropriate entity of the processor that can generate a request to such remote processor socket(s) to attempt to reduce the latency. In turn, home agent 325a couples to a memory controller 330a that in turn couples to a local portion of a system memory. More specifically, via a memory interconnect 365a, memory

controller 330a is coupled to and controls access to a dynamic random access memory (DRAM) 370a.

[0041] For purposes of discussion of embodiments herein, processor socket 310a includes an interface 360a which in an embodiment may be a QPI interface to provide interconnection between first processor socket 310a and second processor socket 310b via a QPI interconnect 380. As further shown, a PCU 350a (which may include uncore frequency control logic as described herein) is coupled to interface 360 to enable communication of frequency control messages between the cores.

[0042] Note that embodiments scale to a system with more than two processors as well. In systems where not all processors are connected to each other (for example, where the processors are arranged as a ring, torus or other non-fully-connected topology), intermediate processors act as a local repeater for frequency coordination requests, so that each processor can receive frequency coordination requests from all other processors in the system. Using an embodiment of the present invention, improved power/performance in multiprocessor platforms when running low-bandwidth latency-sensitive server workloads may be realized.

[0043] FIG. 4B is a block diagram illustrating a flow of communications regarding handling of a memory access request in accordance with an embodiment of the present invention. As seen in FIG. 4B, first home agent 325a (or other coherency controller) receives a read request for a given address (1). This request may be part of a larger read transaction flow of a coherency protocol. To respond to the request, a snoop request (2) is sent to remote processor 310b and a memory access is sent to local memory 365a (3). The snoop request may be sent by home agent 325a or by the cache itself, depending on the implementation. In response, remote processor 310b sends a snoop response (4), and local memory 370a sends data for the read (5). Home agent 325a waits for both responses and resolves any conflicts before continuing with the rest of the flow, which includes sending a read completion to the requester. For optimal operation both responses may be received at home agent 325a at substantially the same time.

[0044] In an embodiment, home agent 325a includes counters to count the number of times and the number of cycles that a read completion was stalled waiting for a response from the remote processor (in other words, (4) occurred after (5)). These counters are read by the local PCU, which compares them with thresholds to determine whether to request an increase in the uncore frequency of remote processor 310b. If so, this request is generated and communicated from one PCU to the other by tunneling over interconnect 380. Note that when the counter counts are less than the associated thresholds for a given duration, the PCU may stop requesting an elevated uncore frequency from the remote processor socket, allowing the remote processor socket to return to a frequency based on its own activity level.

[0045] That is, frequency control logic of the PCU can either receive an indication of high latency from the home agent or can periodically read counter values or other information within the latency detection logic of the home agent to determine that such a latency exists and to take appropriate action, namely communication of a frequency coordination request.

[0046] Embodiments can be implemented in processors for various markets including server processors, desktop processors, mobile processors and so forth. Referring now to FIG. 5, shown is a block diagram of a processor in accordance with an embodiment of the present invention. As shown in FIG. 5, processor 400 may be a multicore processor including a plurality of cores 410a – 410n. In one embodiment, each such core may be of an independent power domain and can be configured to enter and exit active states and/or maximum performance states based on workload. The various cores may be coupled via an interconnect 415 to a system agent or uncore 420 that includes various components. As seen, the uncore 420 may include a shared cache 430 which may be a last level cache. In the embodiment shown, the uncore further includes a home agent 445 including a latency detection logic 446. This logic may include a set of counters used for performing latency detection, including a first counter 447 and a second counter 448. In addition, the uncore may include an integrated memory controller 440, various interfaces 450 and a power control unit 455.

[0047] In various embodiments, power control unit 455 may include a uncore frequency control logic 459 in accordance with an embodiment of the present invention. As described above, this logic acts to determine whether an update to a remote processor's uncore frequency is called for (e.g., based on count values received from the counters of latency detection logic 446) and if so to generate a request for an updated uncore frequency to be communicated to one or more remote processors. In addition this logic may perform calculations responsive to messages received from remote processor sockets to enable a coordinated uncore frequency to be generated and communicated to the other processor sockets. Still further, this logic can receive an incoming request for an uncore frequency update and determine whether the requested uncore frequency update is allowed and if so to effect the change, as described herein.

[0048] With further reference to FIG. 5, processor 400 may communicate with a system memory 460, e.g., via a memory bus. In addition, by interfaces 450, connection can be made to various off-chip components such as peripheral devices, mass storage and so forth. While shown with this particular implementation in the embodiment of FIG. 5, the scope of the present invention is not limited in this regard.

[0049] Referring now to FIG. 6, shown is a block diagram of a multi-domain processor in accordance with another embodiment of the present invention. As shown in the embodiment of FIG. 6, processor 500 includes multiple domains. Specifically, a core domain 510 can include a plurality of cores 510_0 – 510_n , a graphics domain 520 can include one or more graphics engines, and a system agent domain 550 may further be present. In some embodiments, system agent domain 550 may execute at an independent frequency than the core domain and may remain powered on at all times to handle power control events and power management such that domains 510 and 520 can be controlled to dynamically enter into and exit high power and low power states. Each of domains 510 and 520 may operate at different voltage and/or power. Note that while only shown with three domains, understand the scope of the present invention is not limited in this regard and additional domains can be present in other embodiments. For example, multiple core domains may be present each including at least one core.

[0050] In general, each core 510 may further include low level caches in addition to various execution units and additional processing elements. In turn, the various cores may be coupled to each other and to a shared cache memory formed of a plurality of units of a last level cache (LLC) 540₀ – 540_n. In various embodiments, LLC 540 may be shared amongst the cores and the graphics engine, as well as various media processing circuitry. As seen, a ring interconnect 530 thus couples the cores together, and provides interconnection between the cores, graphics domain 520 and system agent circuitry 550. In one embodiment, interconnect 530 can be part of the core domain. However in other embodiments the ring interconnect can be of its own domain.

[0051] As further seen, system agent domain 550 may include display controller 552 which may provide control of and an interface to an associated display. As further seen, system agent domain 550 may include a power control unit 555 which can include an uncore frequency control logic 559 in accordance with an embodiment of the present invention to enable control and/or coordination of uncore frequency in multiple sockets of a multiprocessor system. In various embodiments, this logic may execute the algorithms described above in FIGS. 3A and 3B.

[0052] As further seen in FIG. 6, processor 500 can further include an integrated memory controller (IMC) 570 that can provide for an interface to a system memory, such as a dynamic random access memory (DRAM). In turn, this local memory may be owned by a home agent 545 that includes latency detection logic 546 to determine whether remote processor socket responses received by home agent 545 have a latency greater than a threshold value. Multiple interfaces 580₀ – 580_n may be present to enable interconnection between the processor and other circuitry. For example, in one embodiment at least one direct media interface (DMI) interface may be provided as well as one or more Peripheral Component Interconnect Express (PCI Express™ (PCIe™)) interfaces. Still further, to provide for communications between other agents such as additional processors or other circuitry, one or more interfaces in accordance with an Intel® Quick Path Interconnect (QPI) protocol may also be provided. Although shown at this high level in the embodiment of FIG. 6, understand the scope of the present invention is not limited in this regard.

[0053] Embodiments may be implemented in many different system types. Referring now to FIG. 7, shown is a block diagram of a system in accordance with an embodiment of the present invention. As shown in FIG. 7, multiprocessor system 600 is a point-to-point interconnect system, and includes a first processor 670 and a second processor 680 coupled via a point-to-point interconnect 650. As shown in FIG. 7, each of processors 670 and 680 may be multicore processors, including first and second processor cores (i.e., processor cores 674a and 674b and processor cores 684a and 684b), although potentially many more cores may be present in the processors. Each of the processors can include a PCU, a home agent, and other logic to perform remote processor latency detection and uncore frequency control and coordination, as described herein.

[0054] Still referring to FIG. 7, first processor 670 further includes a memory controller hub (MCH) 672 and point-to-point (P-P) interfaces 676 and 678. Similarly, second processor 680 includes a MCH 682 and P-P interfaces 686 and 688. As shown in FIG. 7, MCH's 672 and 682 couple the processors to respective memories, namely a memory 632 and a memory 634, which may be portions of system memory (e.g., DRAM) locally attached to the respective processors. First processor 670 and second processor 680 may be coupled to a chipset 690 via P-P interconnects 662 and 664, respectively. As shown in FIG. 7, chipset 690 includes P-P interfaces 694 and 698.

[0055] Furthermore, chipset 690 includes an interface 692 to couple chipset 690 with a high performance graphics engine 638, by a P-P interconnect 639. In turn, chipset 690 may be coupled to a first bus 616 via an interface 696. As shown in FIG. 7, various input/output (I/O) devices 614 may be coupled to first bus 616, along with a bus bridge 618 which couples first bus 616 to a second bus 620. Various devices may be coupled to second bus 620 including, for example, a keyboard/mouse 622, communication devices 626 and a data storage unit 628 such as a disk drive or other mass storage device which may include code 630, in one embodiment. Further, an audio I/O 624 may be coupled to second bus 620. Embodiments can be incorporated into other types of systems including mobile devices such as a smart cellular telephone, tablet computer, netbook, Ultrabook™, or so forth.

[0056] Embodiments may be used in many different types of systems. For example, in one embodiment a communication device can be arranged to perform the various methods and techniques described herein. Of course, the scope of the present invention is not limited to a communication device, and instead other embodiments can be directed to other types of apparatus for processing instructions, or one or more machine readable media including instructions that in response to being executed on a computing device, cause the device to carry out one or more of the methods and techniques described herein.

[0057] Embodiments may be implemented in code and may be stored on a non-transitory storage medium having stored thereon instructions which can be used to program a system to perform the instructions. The storage medium may include, but is not limited to, any type of disk including floppy disks, optical disks, solid state drives (SSDs), compact disk read-only memories (CD-ROMs), compact disk rewritables (CD-RWs), and magneto-optical disks, semiconductor devices such as read-only memories (ROMs), random access memories (RAMs) such as dynamic random access memories (DRAMs), static random access memories (SRAMs), erasable programmable read-only memories (EPROMs), flash memories, electrically erasable programmable read-only memories (EEPROMs), magnetic or optical cards, or any other type of media suitable for storing electronic instructions.

[0058] While the present invention has been described with respect to a limited number of embodiments, those skilled in the art will appreciate numerous modifications and variations therefrom. It is intended that the appended claims cover all such modifications and variations as fall within the true spirit and scope of this present invention.

What is claimed is:

- 1 1. A processor comprising:
2 at least one core to execute instructions; and
3 uncore means coupled to the at least one core, the uncore logic including a first
4 means for receiving memory access requests from the at least one core and routing
5 the memory access requests to a local memory coupled to the processor and routing
6 snoop requests corresponding to the memory access requests to at least one remote
7 processor coupled to the processor, wherein the first means is to maintain latency
8 information regarding a difference between receipt of responses to the snoop
9 requests from the at least one remote processor and receipt of responses to the
10 memory access requests from the local memory.
- 1 2. The processor of claim 1, wherein the first means includes a first counter
2 means for maintaining a count of a number of instances in which a response from a
3 remote processor regarding a snoop request was received after a response from the
4 local memory regarding a corresponding memory access request.
- 1 3. The processor of claim 2, wherein the first means is to increment the first
2 counter means per instance.
- 1 4. The processor of claim 2, wherein the first means includes a second counter
2 means for maintaining a count of a number of cycles for the number of instances.
- 1 5. The processor of claim 4, wherein the first means is to update the second
2 counter means for each instance by a number of cycles of delay that the response
3 from the remote processor was received after the response from the local memory.
- 1 6. The processor of claim 4, further comprising a power control means for
2 receiving the latency information, and responsive to the latency information,
3 generating a request for communication to the at least one remote processor to

4 request frequency coordination between the processor and the at least one remote
5 processor.

1 7. The processor of claim 6, wherein the power control means is to read the first
2 counter means and the second counter means and to generate the request based at
3 least on the first and second counter means.

1 8. The processor of claim 1, wherein the processor and the at least one remote
2 processor are of a multiprocessor platform.

1 9. The processor of claim 1, further comprising an uncore frequency control
2 means for receiving a frequency coordination request from the at least one remote
3 processor and updating a frequency for the uncore means responsive to the
4 frequency coordination request.

1 10. The processor of claim 9, wherein the uncore frequency control means is to
2 receive a plurality of frequency coordination requests each from a processor coupled
3 to the processor and to select an updated frequency for the uncore means
4 corresponding to a frequency less than a maximum requested frequency of the
5 plurality of frequency coordination requests by at least one bin frequency.

1 11. A method comprising:
2 receiving a first response in a first logic of a processor responsive to sending a
3 memory access request to a first memory coupled to the processor and receiving a
4 second response in the first logic responsive to sending a snoop request to a remote
5 processor coupled to the processor;
6 updating a first counter and a second counter based on a delay between receipt of
7 the first response and receipt of the second response; and
8 providing data obtained responsive to the memory access request to a requester.

1 12. The method of claim 11, further comprising incrementing the first counter per
2 instance of the delay, the first counter to maintain a count of a number of instances

3 in which a response from the remote processor to a snoop request was received
4 after a corresponding response from the first memory to a memory access request.

1 13. The method of claim 12, further comprising incrementing the second counter
2 for each instance by a number of cycles of the delay, the second counter to maintain
3 a count of a number of cycles for the number of instances.

1 14. The method of claim 11, further comprising responsive to a value of the first
2 counter and a value of the second counter, generating a request for communication
3 to the remote processor to request frequency coordination.

1
2 15. The method of claim 11, further comprising:
3 receiving a frequency update request in the processor from the remote processor;
4 and
5 updating a frequency of an uncore logic of the processor responsive to the frequency
6 update request, wherein the updated frequency is at a value greater than a
7 frequency determined by a power controller of the processor based on a workload
8 being executed by the processor.

1 16. A system comprising:
2 a first processor including at least one core to execute instructions, uncore
3 logic coupled to the at least one core, a power control unit (PCU) coupled to the at
4 least one core to control a power consumption level of the at least one core, the PCU
5 including an uncore frequency control means for receiving a frequency coordination
6 request from a second processor and updating a frequency for the uncore logic
7 responsive to the frequency coordination request, and a communication interface to
8 receive the frequency coordination request from the second processor via an
9 interconnect;
10 the second processor coupled to the first processor via the interconnect and
11 including a second uncore logic, a second core, and a latency detection means for
12 determining a latency between receipt of a response to a snoop request received
13 from the first processor and receipt of a corresponding response to a memory

14 access request received from a second local memory coupled to the second
15 processor, the second processor to generate the frequency coordination request
16 based at least in part on the latency; and
17 a first local memory coupled to the first processor and the second local
18 memory coupled to the second processor.

1 17. The system of claim 16, wherein the second processor further comprises a
2 home agent including the latency detection means to receive the memory access
3 request from the second core and to route the memory access request to the second
4 local memory and to route the snoop request to the first processor.

1 18. The system of claim 16, wherein the latency detection means comprises:
2 a first counter to maintain a count of a number of instances in which a response from
3 the first processor regarding a snoop request was received after a response from the
4 second local memory regarding a corresponding memory access request; and
5 a second counter to maintain a count of a number of cycles for the number of
6 instances.

1 19. The system of claim 18, wherein the latency detection means is to update the
2 first counter per instance and update the second counter for each instance by a
3 number of cycles of delay that the response from the first processor was received
4 after the response from the second local memory.

1 20. The system of claim 16, wherein the second processor further comprises a
2 second uncore frequency control means for receiving information regarding the
3 latency and responsive to the latency information, generating the frequency
4 coordination request.

1 21. The system of claim 16, wherein the first processor is to receive a plurality of
2 frequency coordination requests each from a processor coupled to the first processor
3 and to select an updated frequency corresponding to a frequency less than a
4 maximum requested frequency of the plurality of frequency coordination requests by

5 at least one bin frequency, and to communicate the selected updated frequency to
6 each of the processors.

1 22. The system of claim 16, wherein responsive to the updated uncore logic
2 frequency, a home agent of the second processor is to receive a second response to
3 a second snoop request from the first processor and a corresponding second
4 response to a second memory access request from the second local memory at
5 substantially the same time.

100

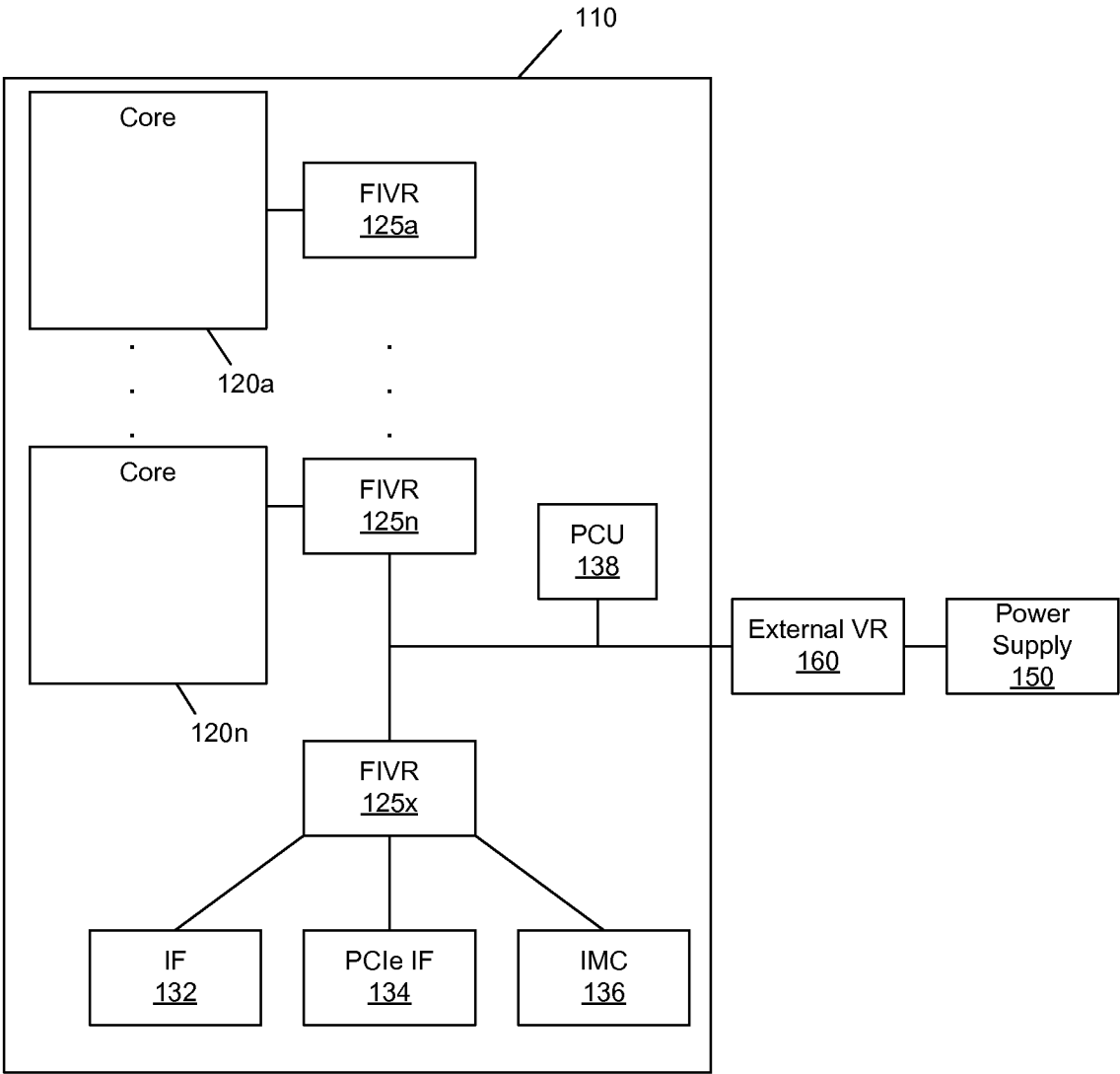


FIG. 1

2/9

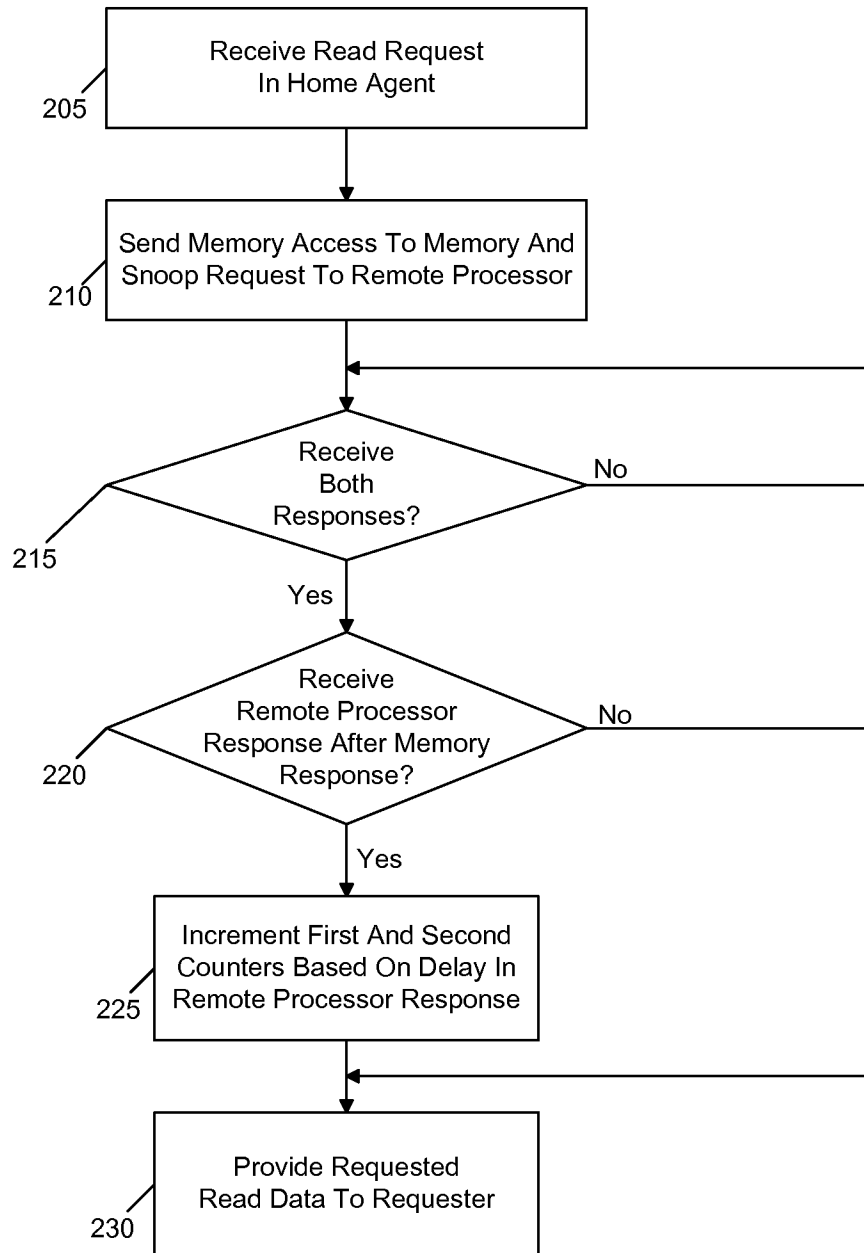
200

FIG. 2

3/9

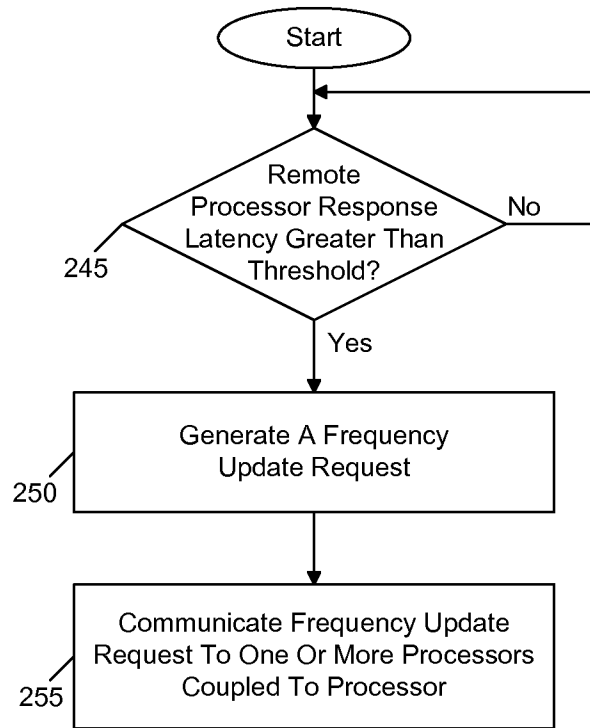
240

FIG. 3A

4/9

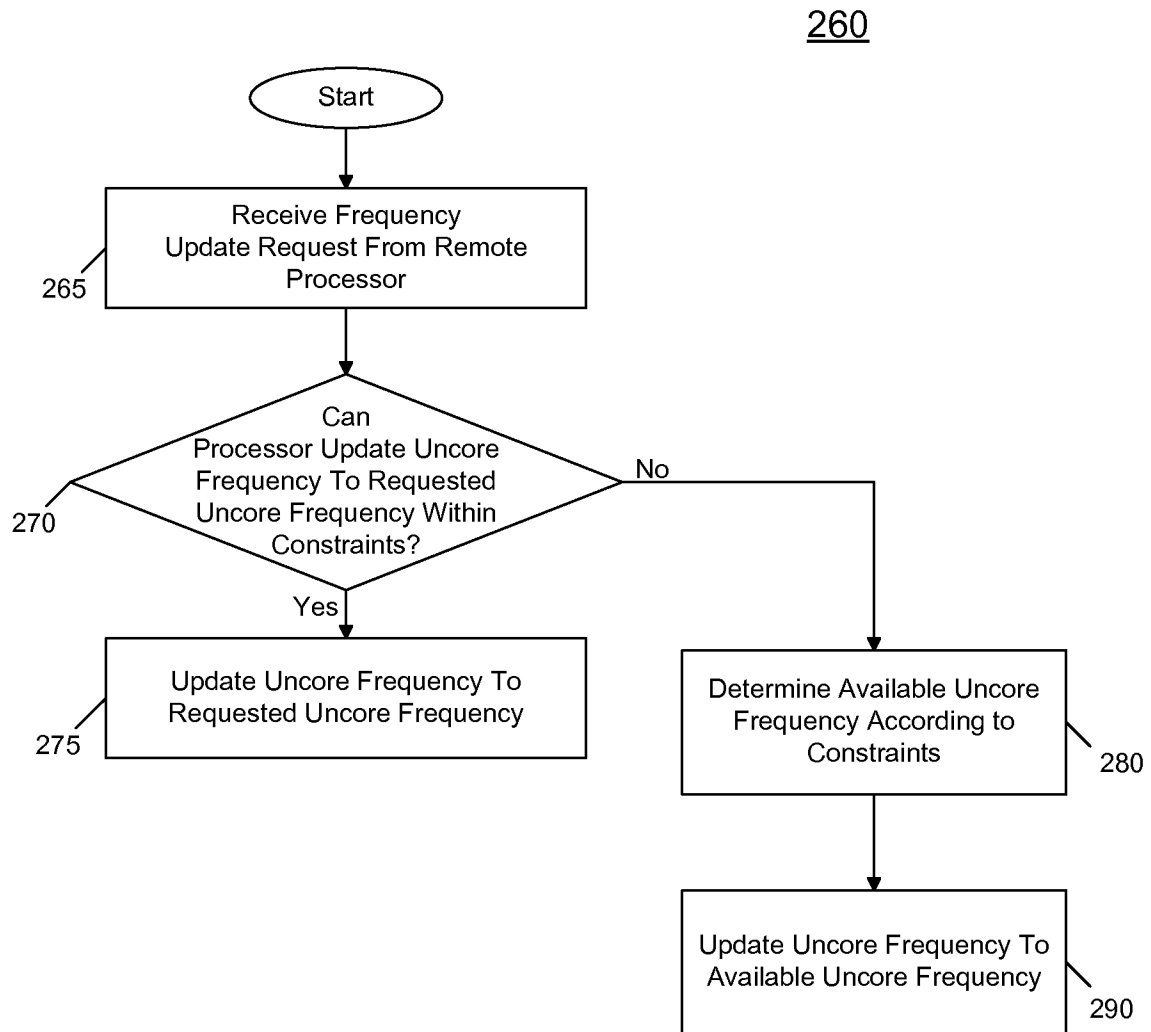


FIG. 3B

300

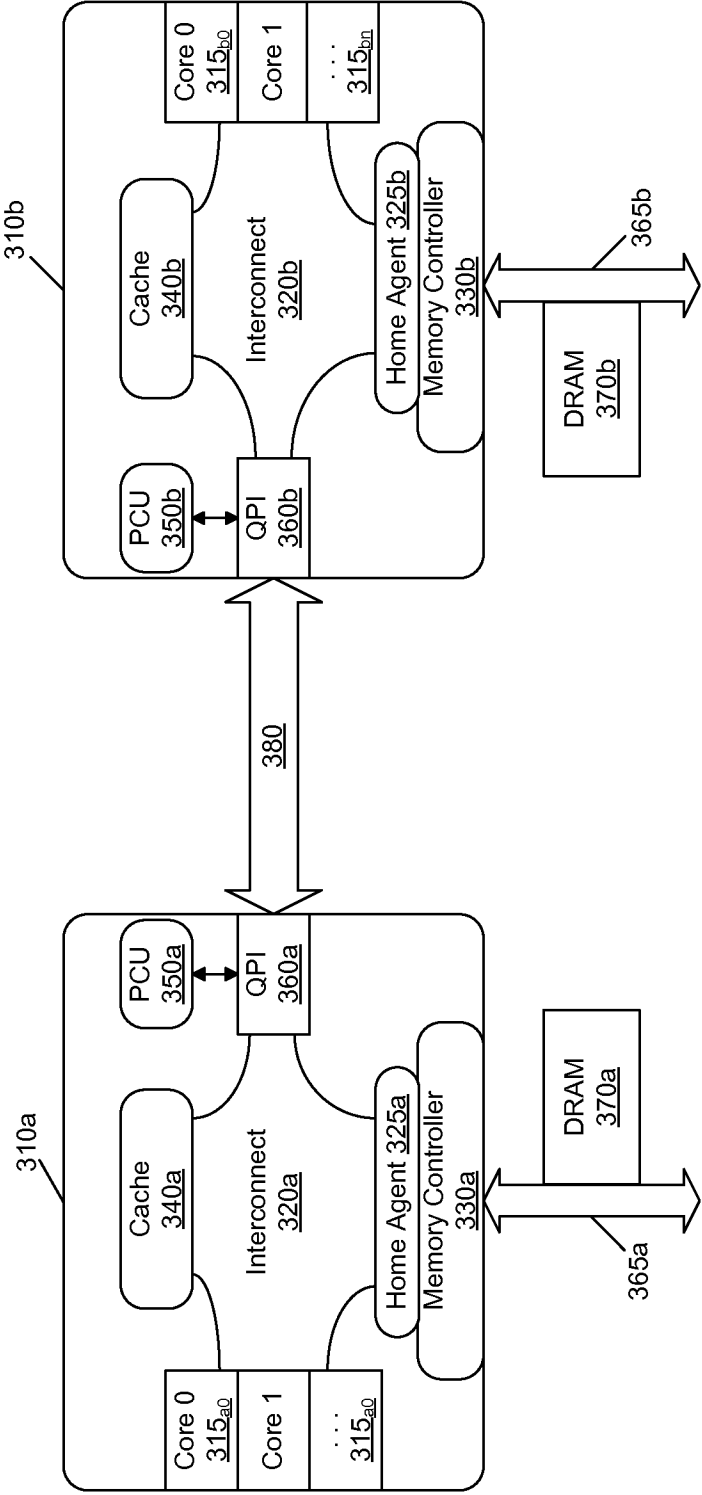


FIG. 4A

300

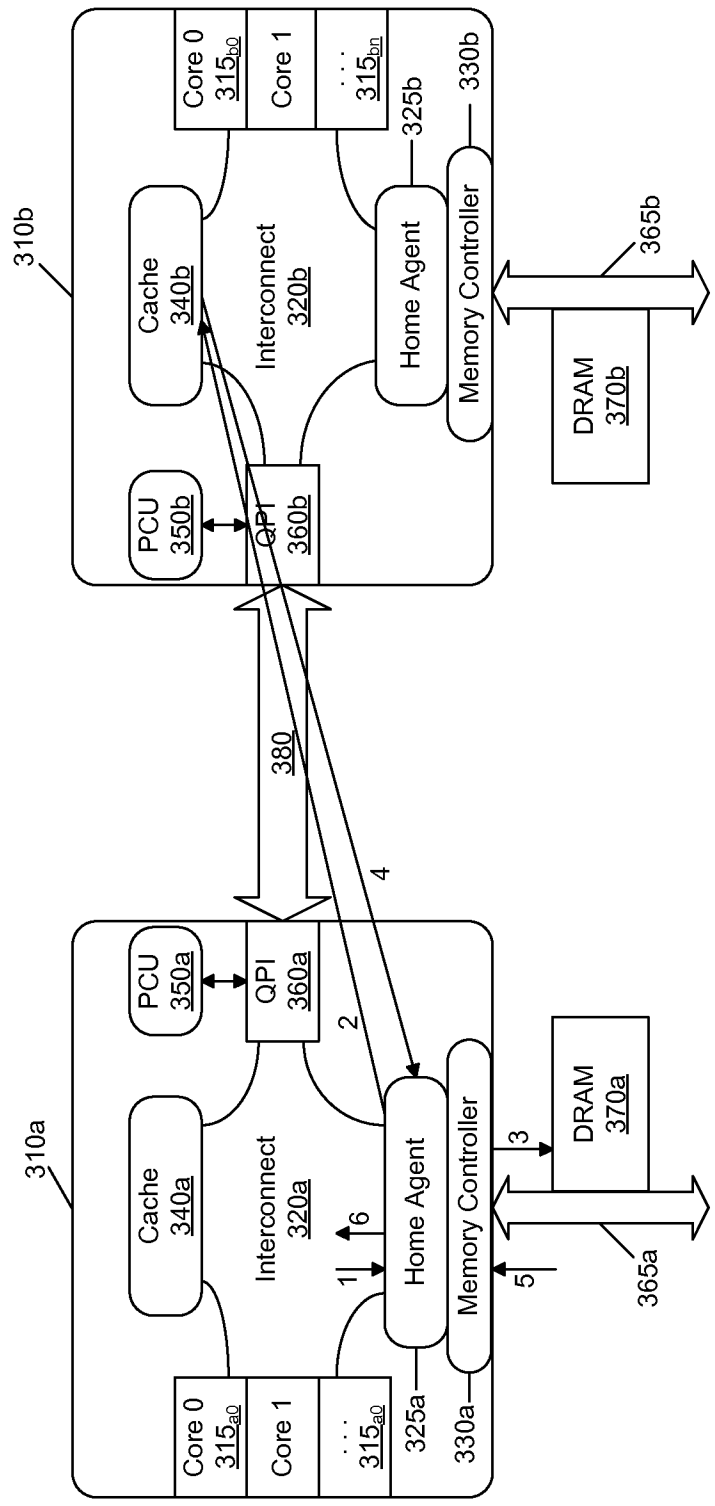


FIG. 4B

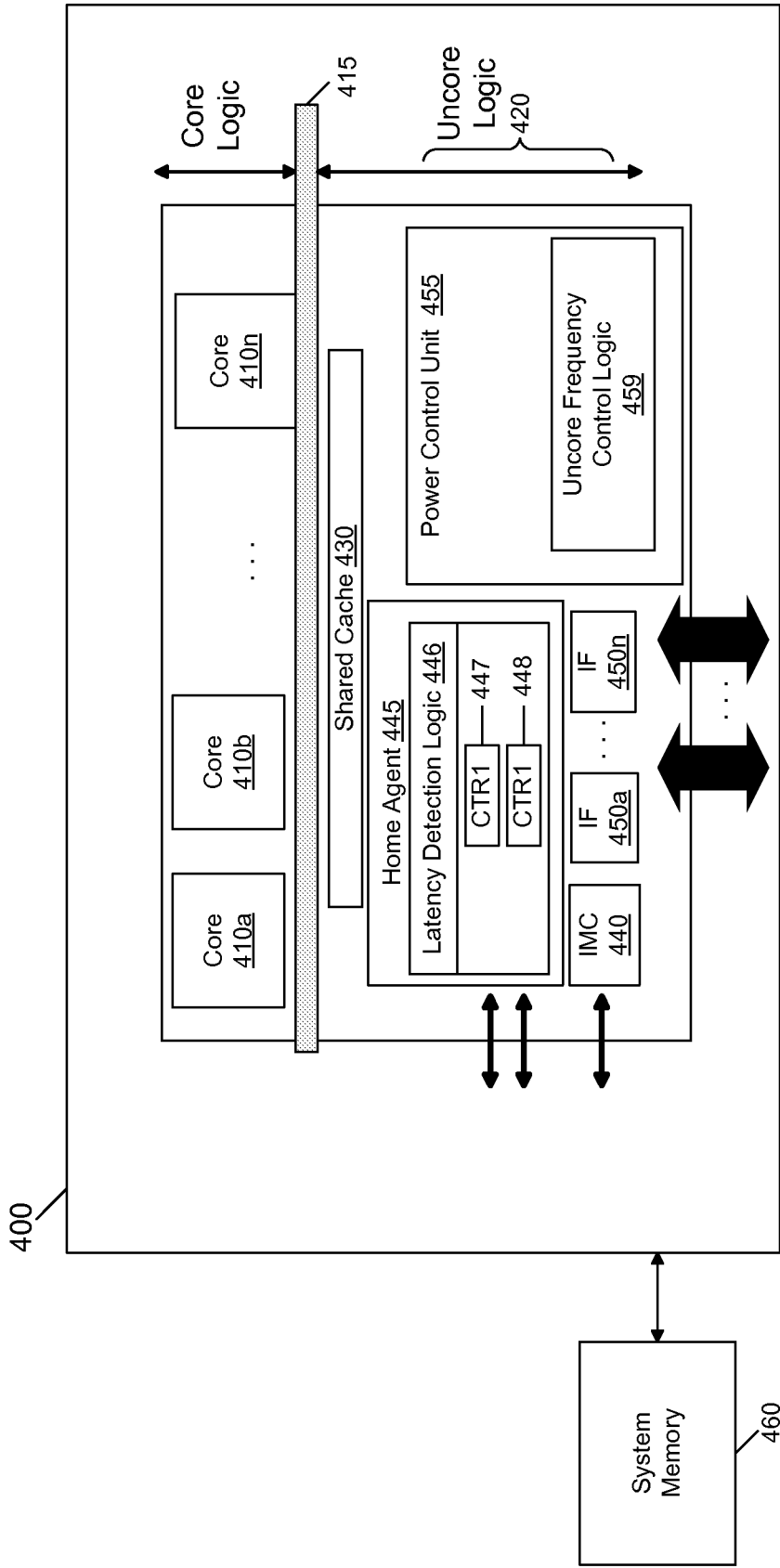


FIG. 5

500

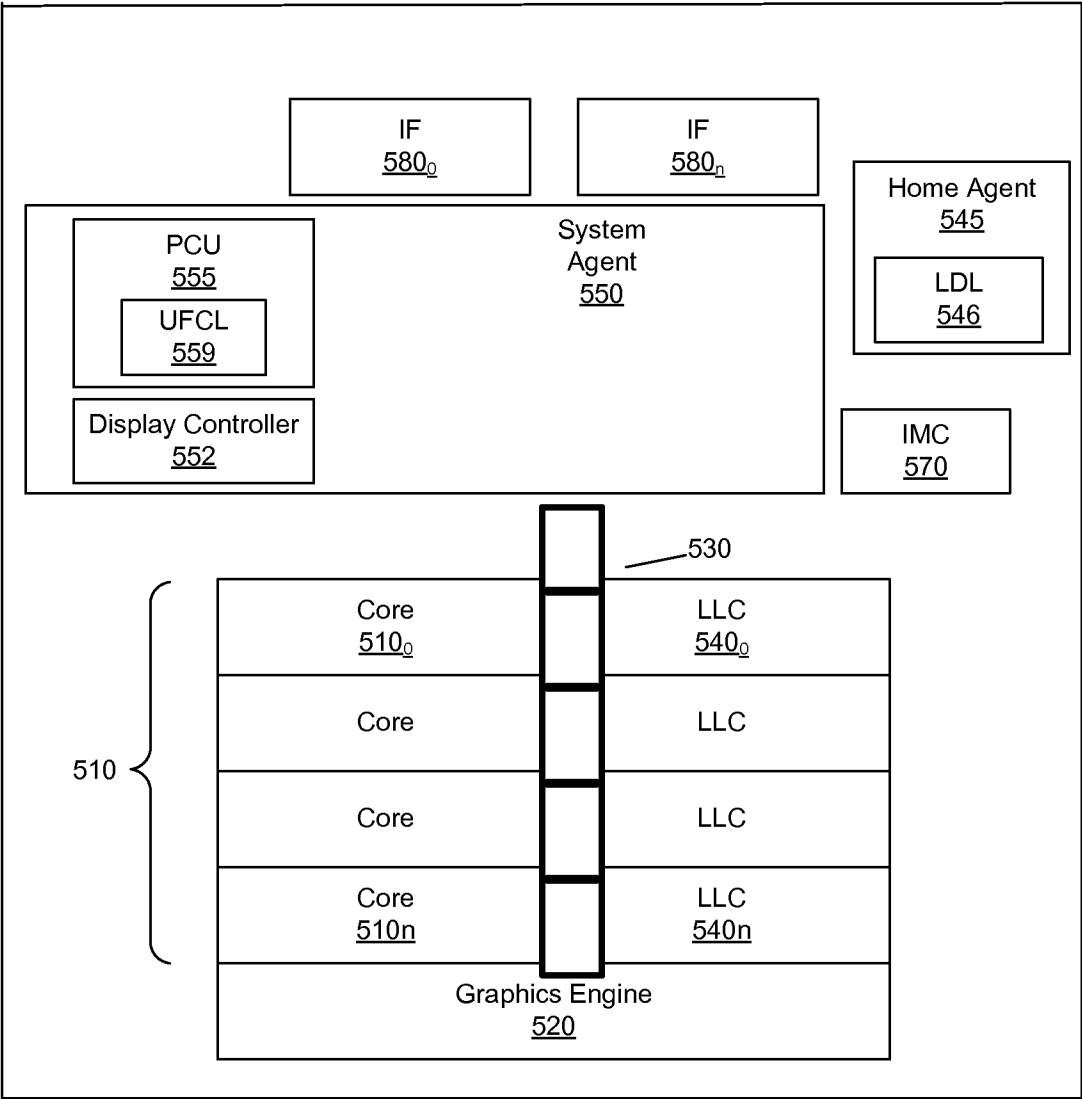


FIG. 6

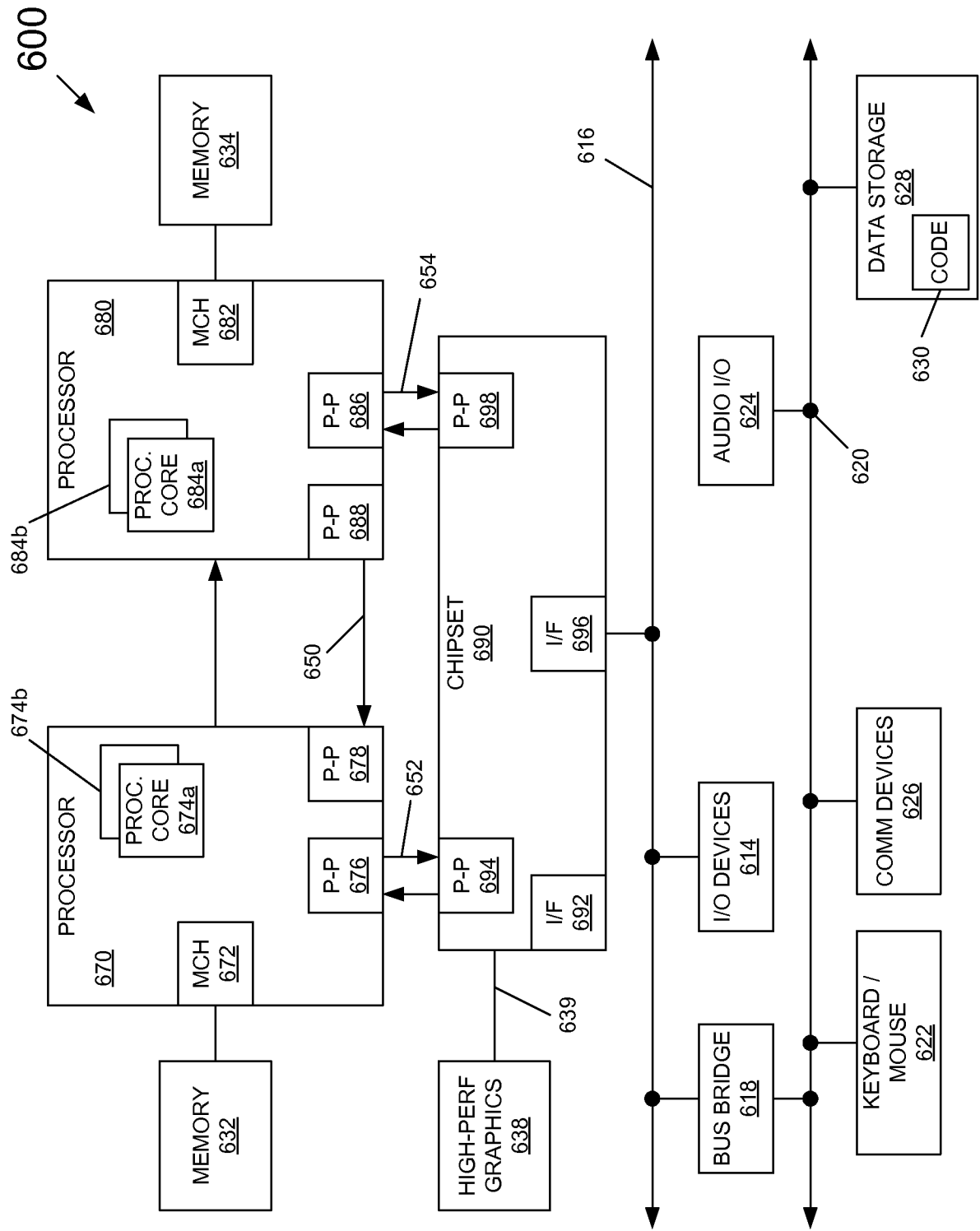


FIG. 7

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2013/048079**A. CLASSIFICATION OF SUBJECT MATTER****G06F 13/14(2006.01)i, G06F 13/16(2006.01)i, G06F 12/00(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 12/00; G06F 1/26; G06F 13/28; G06F 13/16

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: processor, frequency coordination, uncore, memory, access, snoop, latency

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2006-0080508 A1 (HOOVER et al.) 13 April 2006 See paragraph [0034], claims 1, 10-11 and figures 1-3, 5.	1-22
A	US 2012-0079290 A1 (KUMAR et al.) 29 March 2012 See paragraphs [0011], [0034]-[0039], claim 1 and figures 1, 4-5.	1-22
A	US 2007-0180279 A1 (SULLAM) 02 August 2007 See paragraphs [0018]-[0020] and figures 1-2A, 4-5.	1-22
A	US 2006-0179244 A1 (GOODMAN et al.) 10 August 2006 See paragraphs [0044]-[0047], claims 1, 8 and figures 2-3.	1-22
A	US 2008-0005485 A1 (GILBERT et al.) 03 January 2008 See paragraphs [0029]-[0030] and figures 1-3.	1-22



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

25 October 2013 (25.10.2013)

Date of mailing of the international search report

25 October 2013 (25.10.2013)

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

KANG, Sung Chul

Telephone No. +82-42-481-8405



INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2013/048079

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2006-0080508 A1	13/04/2006	US 2007-0294481 A1 US 7305524 B2	20/12/2007 04/12/2007
US 2012-0079290 A1	29/03/2012	CN 103229122 A GB 2498148 A KR 10-2013-0061747 A TW 201218072 A US 2013-185570 A1 WO 2012-040052 A2 WO 2012-040052 A3	31/07/2013 03/07/2013 11/06/2013 01/05/2012 18/07/2013 29/03/2012 31/05/2012
US 2007-0180279 A1	02/08/2007	US 7555664 B2	30/06/2009
US 2006-0179244 A1	10/08/2006	US 2008-0215824 A1 US 7404046 B2 US 7941611 B2	04/09/2008 22/07/2008 10/05/2011
US 2008-0005485 A1	03/01/2008	CN 101097545 A CN 101097545 B DE 102007030116 A1 GB 0712625 D GB 2439650 A GB 2439650 B JP 2008-027435 A JP 4737691 B2 TW 200815992 A TW I336846 A US 7581068 B2	02/01/2008 30/05/2012 14/02/2008 08/08/2007 02/01/2008 28/09/2011 07/02/2008 03/08/2011 01/04/2008 01/02/2011 25/08/2009