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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
3 April 2014 (03.04.2014)

WIPO | PCT

(10) International Publication Number
WO 2014/051615 A1

(51) International Patent Classification:
G06F 9/06 (2006.01) *G06F 1/00* (2006.01)

(21) International Application Number:
PCT/US2012/057869

(22) International Filing Date:
28 September 2012 (28.09.2012)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P.** [US/US]; 11445 Compaq Center Drive W., Houston, Texas 77070 (US).

(72) Inventors: **LABORDE, Louis**; 19410 Homestead Road, Cupertino, California 95014 (US). **FRINK, Spencer A.**; 19410 Homestead Road, Cupertino, California 95014 (US). **GARCIA, Philip**; 19410 Homestead Road, Cupertino, California 95014 (US). **FEVRIER, Thierry**; 2000 Empire Mine Circle, Gold River, California 95670 (US).

(74) Agents: **MCDOWELL, Robert D.** et al.; Hewlett-packard Development Company, L.P., Intellectual Property Administration, Mail Stop 35 3404 E. Harmony Road, Fort Collins, Colorado 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, 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, 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, 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, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

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))*

[Continued on next page]

(54) Title: SYNCHRONIZING TIMESTAMP COUNTERS

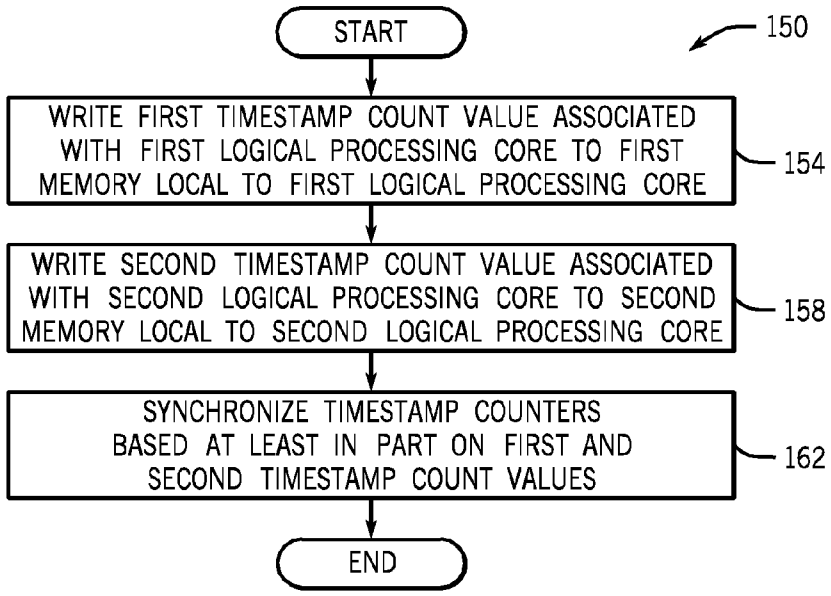

FIG. 2

(57) Abstract: A technique includes writing a first count value associated with a first timestamp counter to a first memory local to the first timestamp counter. The technique includes writing a second count value associated with a second timestamp counter to a second memory local to the second timestamp counter. The timestamp counters are synchronized based at least in part on the first and second count values.

WO 2014/051615 A1

Published:

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

SYNCHRONIZING TIMESTAMP COUNTERS

BACKGROUND

[0001] A typical computer may contain multiple timestamp counters, which may be used for such purposes as tracking the execution of software threads and measuring the timing of events in the computer. The timestamp counters may be incremented by hardware. Generally, timestamp counters provide manners of retrieving CPU timing information.

BRIEF DESCRIPTION OF THE DRAWINGS

[0002] Fig. 1 is a schematic diagram of a computer system according to an example implementation.

[0003] Fig. 2 is a flow diagram depicting a technique to synchronize timestamp counters of a computer according to an example implementation.

[0004] Fig. 3 is a schematic diagram of an apparatus involved in the synchronization of timestamp counters in a computer according to an example implementation.

[0005] Fig. 4 is a flow diagram depicting a technique to use timestamp counters of a computer for purposes of thread execution synchronization according to an example implementation.

[0006] Fig. 5 is a flow diagram depicting a technique to use timestamp counters of a computer for purposes of measuring event timing according to an example implementation.

DETAILED DESCRIPTION

[0008] Techniques and systems are disclosed herein for purposes of synchronizing timestamp counters in a computer. More specifically, systems and techniques are disclosed herein, which account for hardware source(s) of error that may otherwise be present due to asymmetric memory communication paths of the computer.

[0009] As a more specific example, Fig. 1 depicts a "computer," or computer system 10 (a server, a client, a notebook computer, an ultrabook computer, a desktop computer, a tablet computer, a smartphone, and so forth), in accordance with an example implementation. The computer system 10 is a physical machine that is constructed from actual hardware and actual software. In general, the computer system 10 includes multiple, physical processor packages 20 (multi-core processor packages, for example), such as processor packages 20-1 and 20-2, which are depicted in Fig. 1 as examples. It is noted that the computer system 10 may include more than two processor packages 20, in accordance with further implementations.

[0010] In general, the processor package 20 is a semiconductor package, which is constructed to be mechanically and electrically mounted to a motherboard of the computer system 10 via an associated connector, or socket 80. Thus, as depicted in Fig. 1, the processor package 20-1 is illustrated as being connected to a socket 80-1; and the processor package 20-2 is illustrated as being connected to a socket 80-2. In general, the socket 80 is constructed to receive at least a portion of the processor package 20, which contains the package's electrical contacts, and the socket 80 has mechanical features to secure the processor package 20 to the socket 80.

[0011] Thus, the socket 80 contains features to mate with corresponding features of the processor package 20 for purposes of forming the electrical and mechanical connections between the processor package 20 and the motherboard. As a more specific example, in accordance with example implementations, the processor package 20 may be a surface mount package

having a land grid array (LGA) for purposes of forming electrical connections with corresponding pins of the socket 80. Other semiconductor packages may be employed, in accordance with further implementations.

[0012] The processor package 20 contains one or multiple physical processing cores 30, i.e., processing cores that are constructed to execute machine executable instructions. As depicted in the example of Fig. 1, the processor package 20-1 includes multiple processing cores 30; and the processor package 20-1 also contains multiple processing cores 30.

[0013] In accordance with example implementations that are disclosed herein, the computer system 10 employs hyper-threading (HT) technology in which each physical processing core 30 provides an execution "platform" for one or multiple (two, for example) virtual, or logical, processing cores 31. Each physical processing core 30 or logical processing core 31, if implemented, is associated with a timestamp counter 40, which is accessible via a dedicated associated hardware register and which may be a virtual counter or physical counter, depending on the particular implementation.

[0014] In general, hardware of the computer system 10 is constructed to cause the timestamp counters 40 to monotonically increment at the same rate. Machine executable instructions of the computer system 10, and more specifically, an operating system 120 of the computer system 10, undergoes measures to synchronize the count values of the timestamp counters 40. As described further herein, the count values may be synchronized for purposes of allowing the timestamp counters 40 to be used for such purposes as synchronizing operations of the computer 10 and measuring the timing of events occurring in the computer system 10.

[0015] More specifically, although the timestamp counters 40 increment at the same rate, upon the occurrence of certain events of the computer 10, the timestamp counters 40 may be initialized with different count values. For example, when the computer system 10 resets, the timestamp counters 40 come out of reset at different times; and therefore, after a reset, the

timestamp counters 40 may store arbitrary count values. Thus, events such as the computer 10 being reset, powered up, resuming from deep sleep state, or additional processor packages 20 being added to the computer system 10 may cause the timestamp counters 40 to be initialized with various arbitrary count values. Therefore, the operating system 120 undergoes a procedure to synchronize the count values of the timestamp counters 40 so that ideally, after the synchronization, all of the timestamp counters 40 share the same count value in common.

[0016] As an example, in accordance with some implementations, the operating system 120 uses the following count value synchronization technique, although other techniques may be used, in accordance with further implementations. In phase one, the operating system 120 searches for the timestamp counter 40 that has the highest count value. In this manner, the timestamp counter 40 having the highest count value may be designated as a master counter for purposes synchronizing the other counters 40 so that the other counters 40 do not step back in time (due to the highest count value being used) during the synchronization process. To perform the search for the master timestamp counter, in accordance with example implementations, the logical processing cores 31 of the computer system 10 read their timestamp counter registers and write the corresponding read count values to a designated memory location so that the operating system 120 may review the count values to select the timestamp counter 40 having the highest count value to be the master timestamp counter.

[0017] Next, in accordance with example implementations, in phase two of the count value synchronization technique, the operating system 120 pairs each of the remaining timestamp counters 40 (one at a time) with the master timestamp counter to synchronize the count value of the paired timestamp counter 40 to the count value of the master timestamp counter. In phase two, the logical processing cores 31 (one or each counter 40 of the pair, for example) exchange count values associated with their timestamp counters 40, and the operating system 120 sets the count value of the paired

timestamp counter 40 based on the exchanged values. At the conclusion of phase two, the timestamp counters 40 ideally share a common count value. The asymmetric memory communication paths of the computer 10 involved in the count value exchanges may, however, introduce significant errors in the counter synchronization, if not for the systems and techniques that are disclosed herein.

[0018] More specifically, in accordance with example implementations, the computer system 10 employs a non-uniform memory architecture (NUMA), an architecture in which each processor package 20 contains one or more associated memory controllers 60 (one memory controller 60 per package 20 being depicted in Fig. 1, as an example). Therefore, as depicted in Fig. 1, a given processor package 20 uses its associated memory controller(s) 60 for purposes of reading from and writing to memory.

[0019] The inclusion of the memory controller(s) 60 in the processor packages 20 according to the NUMA results in asymmetric memory access times, however. More specifically, the memory that is accessible by a given memory controller 60 may be "local" in that the memory controller 60 may generate read and write requests to target memory space associated with one or multiple memory modules 90 that are local to the associated socket 80. However, some memory accesses by a given memory controller 60 may not be local accesses, as these accesses involve read and write requests that are further processed, as appropriate, by the memory controller 60 of another processor package 20.

[0020] Therefore, for the example of Fig. 1, the processor package 20-1 has associated local memory modules 90-1 (double data rate (DDR) dynamic random access memory (DRAM) modules, as examples), which are not considered local for the processor package 20-2; and the processor package 20-2 has associated local memory modules 90-2, which are not considered local for the processor package 20-1.

[0021] For purposes of accessing non-local memory, the processor package 20 includes an interconnect controller 64 (a Quick Path Interconnect (QPI) controller or a HyperTransport controller, as examples), which responds to non-local read and write requests for purposes of forwarding these requests to the appropriate memory controller or bridge/hub of the computer system 10. In this manner, the interconnect controller 64 may communicate with other non-processor package components of the computer system 10 via an input/output (I/O) hub 100, as well as communicate with the memory controller 60 of another processor package 20.

[0022] The processor package 20 may further include various caches, such as, for example, level one (L1) and level two (L2) caches on each processing core 30 and a level three (L3) cache 50 that is shared among the processing cores 30. Other cache architectures may be employed, in accordance with further implementations.

[0023] Due to the above-described asymmetry in the memory communication paths, accesses to a given memory location may vary among the logical processing cores 31, resulting in different memory access latency times. These different memory access latency times may introduce challenges in properly synchronizing the count values of the timestamp counters 40.

[0024] More specifically, due to the above-described NUMA in which each processor package 20 has its own associated memory controller(s) 60 and thus, has its local memory, exchanging count values using the logical processing cores 31 of the different processor packages 20 may, without the techniques and systems that are disclosed herein, introduce errors due to asymmetric memory communication paths.

[0025] For example, if a given memory location of the computer system 10 is selected for exchanging timestamp count values, then, from a memory access latency time perspective, the memory communication path for a logical processing core 31 of one processor package 20 to the memory location may be "farther" than for a logical processing core 31 of another processor

package 20. Thus, for example, if a memory location in the memory modules 90-2 is hypothetically selected for the exchange, a logical processing core 31 of the processor package 20-2 may effectively store its counter value in an internal cache of the processor package 20-2, whereas a logical processing core 31 of the processor package 20-1, for this example exchange, performs a write that involves communications through two interconnect controllers and the memory controller 60 of the processor package 20-2. Due to the potentially different memory access time differences, such a technique to synchronize the timestamp counters 40 may be relatively sensitive to the memory location that is selected for the exchange.

[0026] Techniques are disclosed herein for purpose of overcoming memory communication path asymmetry issues that may otherwise arise when timestamp count values 54 associated with different processor packages 20 are exchanged. More specifically, in accordance with example implementations that are disclosed herein, local memories are used to exchange the timestamp count values. In this manner, to exchange timestamp count values for a pair of timestamp counters 40, the logical processing core 31 associated with each timestamp counter 40 writes its associated count value 54 to a local memory (relative to the writing logical processing core 31) and reads the other count value from the other (potentially non-local) memory location. Thus, the exchanged timestamp count values are written to local memories (such as local cache lines, as further described herein) and due to this arrangement, the access asymmetries are cancelled.

[0027] In accordance with some implementations, the count value exchange may occur in an ordered sequence. For example, the logical processing core 31 writing the master timestamp count value may first write the count value to local memory, and then (after observing through polling, for example), the logical processing core 31 writing the other timestamp count value may write its other count value to its local memory. This order may be reversed in further implementations.

[0028] The operating system 120, depending on the particular implementation, may select one of the two timestamp count values 54 as the value to be used for the initialized/synchronized count value; may select one of the timestamp count values 54 as a base count value from which a common count value may be derived; or may derive a common count value based on both of the count values 54. In accordance with some implementations, the operating system 120 may select the higher timestamp count value. Moreover, in accordance with example implementations, the operating system 120 may adjust the higher timestamp count value to account for error sources involved in the exchanging of the count values and use this adjusted count value as the common count value used to initialize the corresponding timestamp counters 40.

[0029] Still referring to Fig. 1, as a more specific example, two timestamp counters 40 that are associated with the processor packages 20-1 and 20-2 are to be synchronized. For this synchronization, a logical processing core 31 of the processor package 20-1 reads its associated timestamp count value 54 from the appropriate local timestamp counter register and writes the count value 54 to its local memory, which effectively involves storing the timestamp count value 54 in a cache line of the shared L3 cache 50 of the processor package 20-1. Likewise, in connection with the time count value exchange, a logical processing core 31 of the processor package 20-2 reads its associated timestamp count value 54 from the appropriate local timestamp counter register and writes the count value 54 to a cache line of the shared L3 cache 50 of the processor package 20-2. The logical processing cores 31 further read the other timestamp count values 54 from the non-local memories to complete the exchange.

[0030] Thus, in general, a technique 150 that is depicted in Fig. 2, may be used for purposes of synchronizing timestamp counters in a computer, which has at least first and second logical processing cores that, in turn, have different associated local memories. Referring to Fig. 2 in conjunction with Fig. 1, pursuant to the technique 150, a first timestamp count value that is

associated with a first logical processing core is written (block 154) to a first memory that is local to the first logical processing core. A second timestamp count value that is associated with the second logical processing core is written (block 158) to a second local memory that is local to the second logical processing core. Pursuant to the technique 150, the timestamp counters are synchronized based at least in part on the first and second count values, as depicted in block 162.

[0031] Therefore, referring to an illustration 200 of Fig. 3, in general, two logical processing cores 220 (logical processing cores 220-1 and 220-2 being depicted in Fig. 3) may synchronize timestamp counters 40 associated with the logical processing cores 220 in the following manner. With this example, the logical processing cores 220 are associated with local memory controllers 60 (memory controllers 60-1 and 60-2, being depicted in Fig. 3). In this regard, each logical processing core 220 stores the associated timestamp count value 54 in its local memory 240 (local memories 240-1 and 240-2 being depicted in Fig. 3) and reads the timestamp count value 54 from the local memory 240 associated with the other logical processing core 220. Based on the exchanged count values, the logical processor cores 220 initialize their associated timestamp counters 40 with a common value that is derived from the exchanged count values 54.

[0032] Referring back to Fig. 1, synchronized timestamp counters 40 may be used in a number of different applications. For example, a set of the timestamp counters 40 (two or more than two timestamp counters 40, for example) may be used to synchronize the execution of associated software threads 110 of the computer system 10. In this manner, the threads 110 may be associated with one or more applications that are concurrently being executed by the computer system 10 by multiple logical processing cores 31.

[0033] Threads 110 that share common data may be "synchronized" (i.e., the execution timings of the threads 110 are regulated relative to each other) so that the shared data is protected when one of the synchronized threads is

accessing the data. The threads that are synchronized may be threads executed by the logical processing cores 31 on the same or different processing cores 30 and/or processor packages 20. The execution timing of a given thread 110 may be regulated, for example, by a count value that is stored by an associated timestamp counter 40 for that thread 110.

[0034] Therefore, for the example implementations in which the timestamp counters 40 are used to synchronize software thread execution, a technique 300 that is depicted in Fig. 4 includes using (block 304) local memories to exchange count values to synchronize timestamp counters and using (block 312) the timestamp counters to synchronize software thread execution.

[0035] The timestamp counters 40 may be used for purposes other than synchronizing threads, in accordance with further implementations. For example, in accordance with an exemplary implementation, two timestamp counters 40 may be used to measure the elapsed time between two events in a computer and therefore, may be synchronized to a common value for this purpose. For example, referring to Fig. 1, in accordance with some implementations, a particular logical processing core 31 may read its timestamp counter 40 for purposes of recording a time for a first event, such as, for example, a time when a network packet is received. In response to a later, second event, such as when the network packet's content is copied to a user, another timestamp count value may be recorded. The difference between these two timestamp count values, in turn, represents the duration of time for the computer to process the packet. Without proper counter synchronization, the measured time may be relatively inaccurate.

[0036] Thus, referring to Fig. 5, a technique 400 in accordance with example implementations includes using (block 404) local memories to exchange count values to synchronize timestamp counters and using (block 412) timestamp counters to synchronize event timing measurement.

[0037] While a limited number of examples have been disclosed herein, those skilled in the art, having the benefit of this disclosure, will appreciate

numerous modifications and variations therefrom. It is intended that the appended claims cover all such modifications and variations.

What is claimed is:

1. A method comprising:

writing a first count value associated with a first timestamp counter to a first memory local to the first timestamp counter, the first timestamp counter being associated with a first memory controller;

writing a second count value associated with a second timestamp counter to a second memory local to the second timestamp counter, the second timestamp counter being associated with a second memory controller other than the first memory controller; and

synchronizing the first and second timestamp counters based at least in part on the first and second count values.

2. The method of claim 1, wherein synchronizing the counters comprises synchronizing the counters to synchronize execution of threads.

3. The method of claim 2, further comprising executing the first thread on a first processor associated with a first processor socket and executing the second thread on a second processor associated with a second processor socket.

4. The method of claim 1, wherein writing the first count value comprises storing the first count value in a cache line.

5. The method of claim 1, wherein writing the first count value comprises storing the first count value inside a first processor package and writing the second count value comprises storing the second count value inside a second processor package other than the first processor package.

6. The method of claim 1, wherein writing the first and second count values comprises writing the first and second count values to memory locations of a non-uniform memory architecture (NUMA)-based computer.

7. The method of claim 1, wherein synchronizing the counters comprises synchronizing counters associated with measuring a timing of events occurring in a computer.

8. An apparatus comprising:

a first logical processing core to write a first count value associated with a first timestamp counter to a first memory local to the first timestamp counter; and

a second logical processing core other than the first logical processing core to write a second count value associated with a second timestamp counter to a second memory local to the second thread,

wherein the first and second logical processing cores are adapted to synchronize the timestamp counters based at least in part on the first and second count values.

9. The apparatus of claim 8, wherein the first logical processing core is associated with a memory controller local to the first memory and the second logical processing core is associated with another memory controller, the another memory controller being local to the second memory.

10. The apparatus of claim 8, wherein the first logical processing core is adapted to read the second count value from the second memory.

11. The apparatus of claim 8, wherein the first logical processing core is associated with a first processor socket and the second processor is associated with a second processor socket other than the first processor socket.

12. The apparatus of claim 8, wherein the first and timestamp counters are used for at least one of measuring a timing of events in a computer and synchronizing thread execution in the computer.

13. An article comprising a non-transitory computer readable storage medium to store instructions that when executed by a computer cause the computer to:

write a first count value associated with a first timestamp counter to a first memory local to the first timestamp counter, the first timestamp counter being associated with a first memory controller;

write a second count value associated with a second timestamp counter to a second memory local to the second timestamp counter, the second timestamp counter being associated with a second memory controller other than the first memory controller; and

synchronize the first and second timestamp counters based at least in part on the first and second count values.

14. The article of claim 13, the storage medium storing instructions that when executed by the computer cause the computer to write the first count value to the first memory and read the second count value from the second memory.

15. The article of claim 13, the storage medium storing instructions that when executed by the computer cause the computer to synchronize the counters in connection with at least one of synchronizing the measuring of an event timing in the computer and synchronizing the execution of threads in the computer.

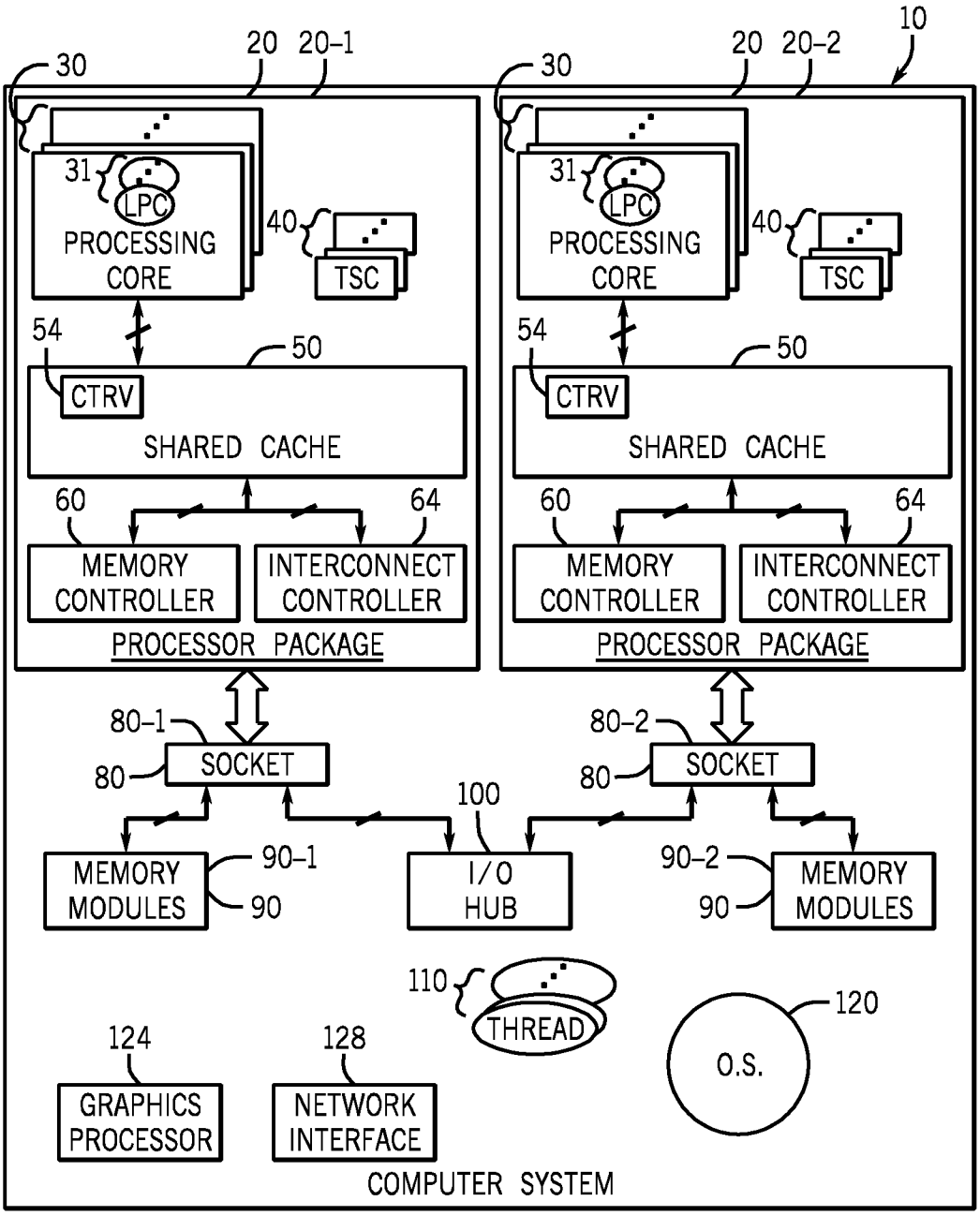
10
20
20-1
30
31
LPC
PROCESSING CORE
40
TSC
54
50
CTRV
SHARED CACHE
60
64
MEMORY CONTROLLER
INTERCONNECT CONTROLLER
PROCESSOR PACKAGE
20
20-2
30
31
LPC
PROCESSING CORE
40
TSC
54
50
CTRV
SHARED CACHE
60
64
MEMORY CONTROLLER
INTERCONNECT CONTROLLER
PROCESSOR PACKAGE
80-1
80
SOCKET
80-2
80
SOCKET
100
MEMORY MODULES
90-1
90
I/O HUB
90-2
90
MEMORY MODULES
110
THREAD
120
O.S.
124
GRAPHICS PROCESSOR
128
NETWORK INTERFACE
COMPUTER SYSTEM

FIG. 1

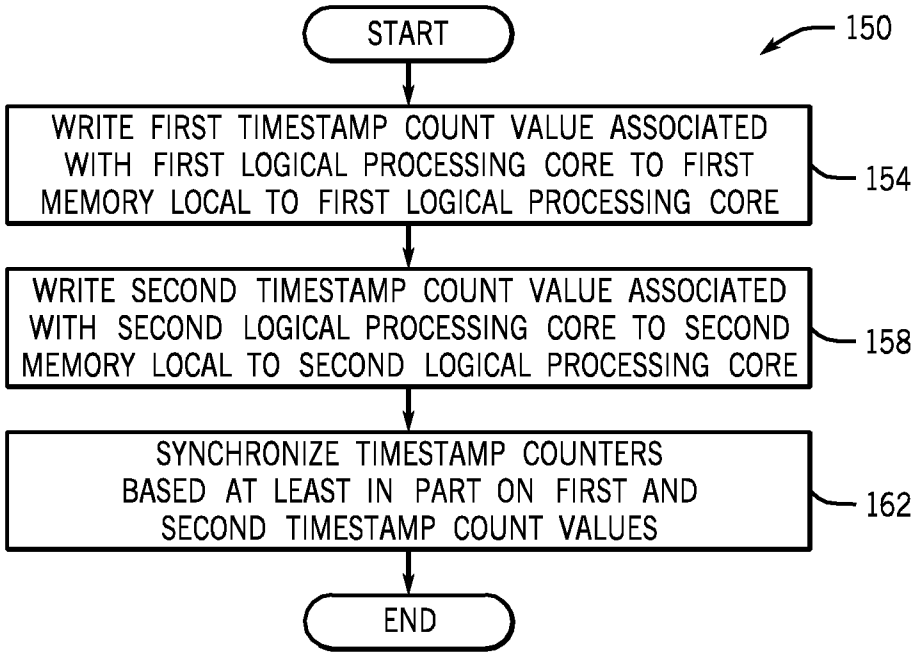
START
150
WRITE FIRST TIMESTAMP COUNT VALUE ASSOCIATED WITH FIRST LOGICAL PROCESSING CORE TO FIRST MEMORY LOCAL TO FIRST LOGICAL PROCESSING CORE
154
WRITE SECOND TIMESTAMP COUNT VALUE ASSOCIATED WITH SECOND LOGICAL PROCESSING CORE TO SECOND MEMORY LOCAL TO SECOND LOGICAL PROCESSING CORE
158
SYNCHRONIZE TIMESTAMP COUNTERS BASED AT LEAST IN PART ON FIRST AND SECOND TIMESTAMP COUNT VALUES
162
END

FIG. 2

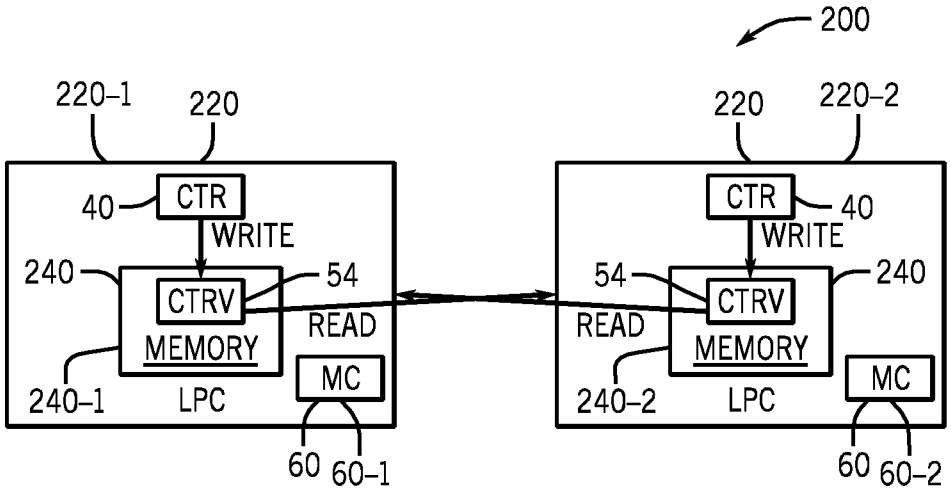
200
220-1
220
220
220-2
40
CTR
WRITE
CTR
40
WRITE
240
54
CTRV
MEMORY
READ
54
CTRV
READ
MEMORY
240
240-1
LPC
MC
240-2
LPC
MC
60
60-1
60
60-2

FIG. 3

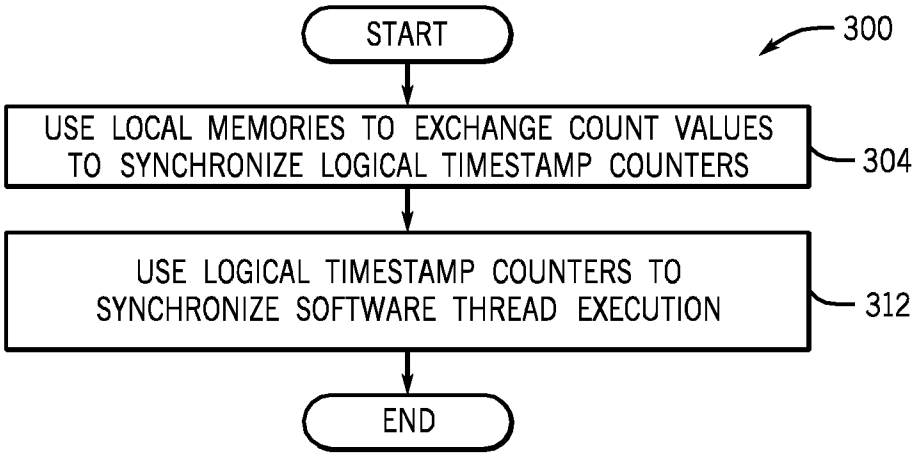
START
300
USE LOCAL MEMORIES TO EXCHANGE COUNT VALUES TO SYNCHRONIZE LOGICAL TIMESTAMP COUNTERS
304
USE LOGICAL TIMESTAMP COUNTERS TO SYNCHRONIZE SOFTWARE THREAD EXECUTION
312
END

FIG. 4

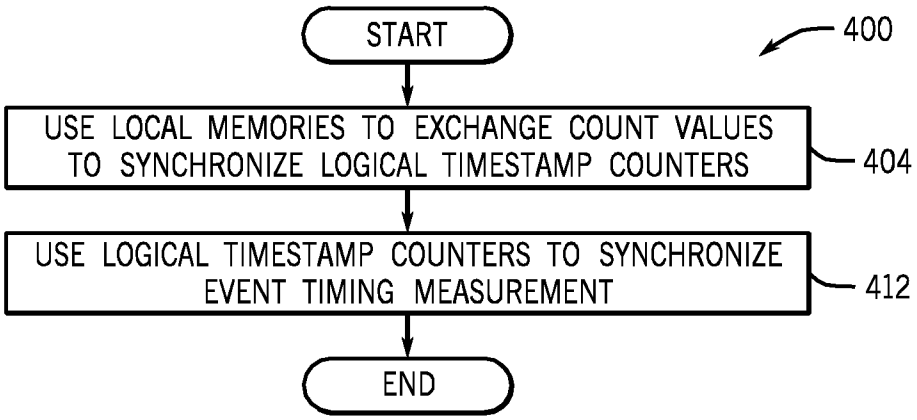
START
400
USE LOCAL MEMORIES TO EXCHANGE COUNT VALUES TO SYNCHRONIZE LOGICAL TIMESTAMP COUNTERS
404
USE LOGICAL TIMESTAMP COUNTERS TO SYNCHRONIZE EVENT TIMING MEASUREMENT
412
END

FIG. 5

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2012/057869

A. CLASSIFICATION OF SUBJECT MATTER

G06F 9/06(2006.01)i, G06F 1/00(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 9/06; G06F 3/00; G06F 1/00; G06F 1/12; G01F 1/00

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: synchronizing, time, stamp, counter, controller, thread, cache

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2009-0222683 A1 (SEREBRIN BENJAMIN C. et al.) 03 September 2009 See paragraphs [0019]-[0024] and claims 1-3	1-15
A	US 2006-0009927 A1 (CHRISTOPHER OSTERLOH et al.) 12 January 2006 See paragraphs [0015]-[0017] and claims 1-5	1-15
A	US 2004-0117682 A1 (JIANZHONG XU) 17 June 2004 See paragraphs [0015]-[0022] and claims 1-9	1-15
A	US 2004-0024925 A1 (ROBERT, E. CYPHER et al.) 05 February 2004 See paragraphs [0086]-[0095] and claims 1-7	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	Date of mailing of the international search report
26 April 2013 (26.04.2013)	**29 April 2013 (29.04.2013)**

Name and mailing address of the ISA/KR

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

Facsimile No. 82-42-472-7140

Authorized officer

KIM, Yong Woong

Telephone No. 82-42-481-5585

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2012/057869

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2009-0222683 A1	03.09.2009	US 7941684 B2	10.05.2011
US 2006-0009927 A1	12.01.2006	US 7343255 B2	11.03.2008
US 2004-0117682 A1	17.06.2004	US 7058838 B2	06.06.2006
US 2004-0024925 A1	05.02.2004	None	