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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
20 April 2006 (20.04.2006)

PCT

(10) International Publication Number
WO 2006/040689 A1

(51) International Patent Classification:
G06F 12/12 (2006.01) *G06F 12/08* (2006.01)

(21) International Application Number:
PCT/IB2005/003565

(22) International Filing Date: 14 July 2005 (14.07.2005)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
10/891,796 15 July 2004 (15.07.2004) US

(71) Applicants *(for all designated States except US)*: **SONY COMPUTER ENTERTAINMENT INC.** [JP/JP]; 2-6-21 Minami-Aoyama, Minato-ku, Tokyo 107-0062 (JP). **INTERNATIONAL BUSINESS MACHINES CORPORATION** [US/US]; New Orchard Road, Armonk, NJ 10504 (US).

(72) Inventors: **OKAWA, Yasukichi**; 7-15-28 Shukugawara Tama-ku, Crea-Persimmon, Apt. 203, Kawasaki 214-0021 (JP). **KIM, Roy, Moonseuk**; 9904 Pickfair Drive, Austin, TX 78750 (US). **LIU, Peichun, Peter**; 9220 Limoncillo Drive, Austin, TX 78750 (US). **TRUONG, Thuong, Quang**; 12612 Picket Rope Lane, Austin, TX 78727 (US).

(74) Agent: **DERNIER, Matthew, B.**; Kaplan & Gilman Gibson & Dernier L.L.P., 900 route 9 North, Woodbridge, NJ 07095 (US).

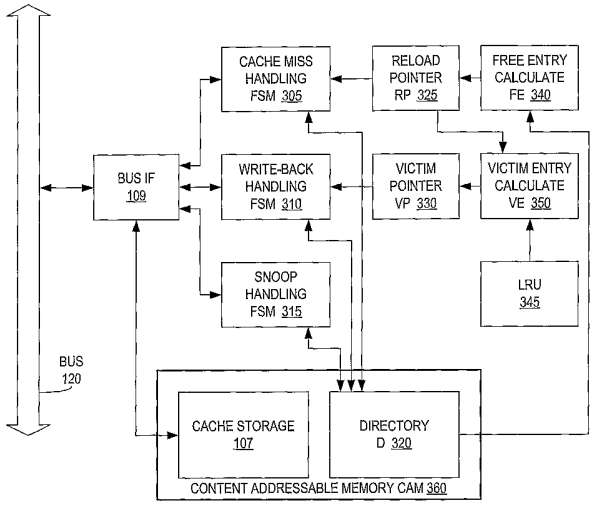
(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KP, KR, KZ, LC, LK,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

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

Published:
— *with international search report*
— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

[Continued on next page]

(54) Title: IMPLEMENTATION AND MANAGEMENT OF MOVEABLE BUFFERS IN CACHE SYSTEM



(57) Abstract: The present invention provides parallel processing of write-back and reload operations in a cache system and optimum circuit utilization by implementing moveable buffers in a cache storage. However, the data and associated pointers are not permanently assigned to a particular buffer - hence, the buffers can move logically around in the facility. Reload pointer is pointing to an empty entry so that retrieved data from the main memory or equal hierarchy cache on cache miss can be always be accommodated. Victim pointer is always pointing to a modified entry for the next candidate of write-back operation. Write-back operation is necessary with reload operation in order to make a free entry for further cache miss handling unless free entry exists. Because of these moveable pointers for reload buffer and victim buffer and integrated write-back buffer in the cache, intra cache data movement is not necessary which improves cache miss handling performance.

WO 2006/040689 A1

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

IMPLEMENTATION AND MANAGEMENT OF MOVEABLE BUFFERS IN CACHE SYSTEM

TECHNICAL FIELD

The present invention relates generally to the field of computer systems and, more particularly, cache buffers.

BACKGROUND

The need for faster computer systems has led to increased demands for high-speed data fetches and stores. A cache system, which is a small, contents addressable memory, with relatively low access latency and high bandwidth, was introduced to meet these requirements.

In a write-back cache system, data modification due to a store instruction is only for the cache. Later on, such modified data write-back cache to the main memory when there is no space to accommodate reloaded data from main memory to resolve cache miss.

Therefore, in order to resolve cache miss when cache is without a free entry, the system uses two distinct operations. One is reload which retrieves demanded data from main memory and allocate it in the cache. Another is write-back cache that writes modified data from victim entry to memory in order to allocate a free entry for a reload operation. Essentially, the reload operation is unable to start as long as write-back is pending.

A conventional write-back cache system accommodates a write-back buffer, where the write-back data moves immediately after the write-back operation initiates. In this manner, write-back operation can employ the write-back buffer so that the reload operation can start utilizing victim entry immediately.

Such a write-back buffer is extra data storage outside cache system, and makes cache design difficult in terms of area

and power consumption.

Therefore, there is a need for a write-back cache system that addresses at least some of the problems associated with conventional write-back cache systems.

SUMMARY OF THE INVENTION

Methods for managing write-back and reload operations in a cache system. Then, employing a plurality of pointers and moveable buffers for receiving storage access instructions in a cache system from one or more processors. The buffers are integrated in the data array and available for reload and write-back operations. A cache controller further reserves a specified reload buffer for cache misses and write-back the victim to memory to keep the reload buffer clear for the next missed entry.

BRIEF DESCRIPTION OF THE DRAWINGS

For a more complete understanding of the present invention, and the advantages thereof, reference is now made to the following Detailed Description taken in conjunction with the accompanying drawings, in which:

FIGURE 1A illustrates an exemplary conventional four-way set associative write-back cache;

FIGURE 1B illustrates an exemplary conventional operational flow of cache replacement;

FIGURE 1C illustrates an exemplary improved process for operational flow of cache replacement;

FIGURE 2 illustrates an exemplary processor cache system interface diagram; and

FIGURE 3 illustrates an exemplary cache system block diagram.

DETAILED DESCRIPTION

In the following discussion, numerous specific details are

set forth to provide a thorough understanding of the present invention. However, those skilled in the art will appreciate that the present invention may be practiced without such specific details. In other instances, well-known elements have been illustrated in block diagram form in order not to obscure the present invention in unnecessary detail. Additionally, for the most part, details concerning network communications, electro-magnetic signaling techniques, and the like, have been omitted inasmuch as such details are not considered necessary to obtain a complete understanding of the present invention, and are considered to be within the understanding of persons of ordinary skill in the relevant art.

It is further noted that, unless indicated otherwise, all functions described herein may be performed in either hardware or software, or some combination thereof. In a preferred embodiment, however, the functions are performed by a processor, such as a computer or an electronic data processor, in accordance with code, such as computer program code, software, and/or integrated circuits that are coded to perform such functions, unless indicated otherwise.

Turning to FIGURE 1A disclosed is an exemplary conventional four-way set associative write-back CACHE 107. A conventional write-back cache needs replacement when a cache miss occurs and there is no empty room in its congruence class. A congruence class set is a set of cache entries indexed by the same index. The cache miss is detected at Index = i. This congruence class has no empty slot. When the victim entry is chosen, and evicted, new data is reloaded, and the cache miss is resolved for the replacement reload has to follow write-back.

Turning to FIGURE 1B, disclosed is an exemplary conventional operational flow of cache replacement. Shown here, two consecutive memory operations are necessary to conduct cache replacement. A conventional cache introduces a

Write-Back Buffer 106 to handle both operations in parallel.

First, a program or a device makes an instruction Request 102, to processor CPU1 105. The instruction goes to a Cache 107 where it is compared by a tag (unique identifier) to the stored tag placed into Cache 107. If there is a match, the data access operation is operated within the cache. If not, a cache miss is recorded and the reload operation is initiated to reload new data to an empty (invalid) entry. If there is no empty entry, and victim calculation logic points to the modified state entry, then modified state entry is castout as a "victim" to Write-Back Buffer 106. Data writes to Main System Memory 140 when bus and main system memory is available. That is, there is only 'n' number of available cachelines, and therefore, victim data must be pushed out to make room for the incoming data that arrives via Bus 120. Bus 120 places the new data into the victim entry line. Reload and write-back are main memory transfer operations that can result in slow transfer and high capacity utilization rates.

The Write-Back Buffer 106 is normally implemented by latch, flip-flop, or even small register file. It is also possible to implement the Write-Back Buffer 106 in the cache 107. Furthermore, when the area per bit and power per bit are large using the latch or flip-flop, it is advantageous to implement the equivalent function with smaller area and power consumption by integrating the Write-Back Buffer 106 into the cache data store (e.g., the SRAM itself). Instead of having a separate Write-Back Buffer 106 inside the cache array, a Reload Pointer 140 is added in the cache array to point to movable reload entry in the cache (delineated further in FIGURE 1C). Victim entry gets a write-back to memory without moving it into a temporary Write-Back Buffer 106 since an empty slot or Reload entry, is always available for concurrent reload. In addition, Reload Pointer 140 moves around in the cache to an available empty slot created by write-back to prevent internal cache

movement of data. If the Reload Pointer 140 is fixed to one location then the reload data has to be moved to another location before the next reload.

Turning now to FIGURE 1C, illustrated is an exemplary improved process for operational flow of cache replacement. Here, a separate write-back buffer is eliminated. Within Cache 107, is at least one open slot in the cache array, coupled logically as a Reload entry.

When a cache miss occurs, and there is only one free entry, a new Victim is selected by victim pointer calculation logic. Then, New Data 103 is loaded in the reload buffer, simultaneously evicting the victim to a Bus 120 if the victim data has been updated with respect to Main System Memory. As soon as that operation completes, the Reload Pointer 140 gets updates.

Turning to FIGURE 2, disclosed is an exemplary processor cache SYSTEM 100 interface diagram. CPU1 105 and CPU2 110 store and retrieve indicia (data, commands, etc.) through their respective caches, 107, and 112, via a typical bus structure. Though there are two processors described here, operating in parallel, and without an apparent master/slave relationship, there can be 'n' number of processors in any configuration, with the same result. The bus interface units, BusIF 109 and BusIF 114 handle main memory requests from the cache system.

The cache systems 107 and 112 receive storage operations requests from processors, and access cache storages accordingly. If there is a cache miss in the cache system 107, for example, the cache system sends a request to BusIF 109 to access Main System Memory 140, or other cache in equal hierarchy to resolve cache miss. If there are 'n' processors with 'x' cache misses occurring simultaneously (either sequentially or in parallel), memory controller MEM CTL 130 determines and queues up the most urgent miss input/output. If there is no empty room in the cache storage to locate retrieved

data for a cache miss, the cache system initiates a write-back request for victim entry to write victim data back to Main System Memory 140.

Turning to FIGURE 3, disclosed is an exemplary cache system block diagram. Within this embodiment are three independent finite state machines (FSMs). Other embodiments can contain more or less FSMs.

FSM 305 handles cache misses. FSM 310 handles write-backs, and FSM 315 accepts and processes snoop requests from other devices hooked on the bus. There are two data pointers. RP 325 is the reload pointer for cache miss handling through FSM 305, and VP 330 is the victim pointer for write-back handling through FSM 310. Cache entry pointed by RP 325 has to be maintained in an empty condition whenever a cache miss occurs because retrieved data for the cache miss is located at the entry pointed by to by RP 325. If there is no free entry in the cache storage 107, (except an entry pointed at by RP 325 on need for cache miss handling), a write-back request will initiate for entry pointed to by VP 330. RP 325 is maintained to point at free entry by free entry calculation FE 340. After write-back is completed, RP 325 is updated by the value of the victim entry, since the victim entry is invalidated by the write-back request. This reload pointer maintenance then prepares for the next cache miss. VP 330 also updates by the output of victim pointer calculation in VP 330 to prepare for the next write-back request (in many instances, the Least Recently Used (LRU) algorithm also calculates for the VP 330 location). Since cache miss data is directly located into the entry pointed to by RP 325 and write-back data is written back directly from entry pointed by VP 330, unnecessary intra-cache data movement from victim entry to write-back entry can be avoided, improving performance and simplifying archive design. There is one directory, D 320 with corresponding cache storage area 107. The directory and cache are coupled, resulting in a

Content Addressable Memory, CAM 360. Directory D 320 is for the storage of tag and cache states for data in corresponding cache storage locations. A tag is the information by which the target address can be associated with a particular directory. The cache state is the data attribute of cache entry to maintain cache coherency among multi-processor system connected via a single bus system. All cache systems must maintain overall cache coherency in terms of cache coherency protocol. Cache-miss finite state machine FSM 305, write-back finite state machine FSM 310, and snoop finite state machine FSM 315 communicate with directories, such as D 320, to retrieve information for target cache entry and to update cache state coherently. VE 350 gets information from a LRU 345 to calculate VP 330. BusIF 109 is the interface to bus 120 (where BUS 120 may be a system bus, a memory bus, Southbridge or other indicia communication pathway). All three finite state machines communicate through BusIF 109, sending and receiving requests through BUS 120.

A snoop request from the BusIF 109 initiates FSM 315 to begin work on a snoop command. BusIF 109 also handles data transfer between cache storage 107 and BUS 120 in accordance with request from one or more of the three finite state machines.

It is understood that the present invention can take many forms and implementations. Accordingly, several variations may be made in the foregoing without departing from the spirit or the scope of the invention. The capabilities outlined herein allow for the possibility of a variety of design and programming models. This disclosure should not be read as preferring any particular design or programming model, but is instead directed to the underlying mechanisms on which these design and programming models can be built.

Having thus described the present invention by reference to certain of its salient characteristics, it is noted that the

features disclosed are illustrative rather than limiting in nature and that a wide range of variations, modifications, changes, and substitutions are contemplated in the foregoing disclosure and, in some instances, some features of the present invention may be employed without a corresponding use of the other features. Many such variations and modifications may be considered desirable by those skilled in the art based upon a review of the foregoing description. Accordingly, it is appropriate that the appended claims be construed broadly and in a manner consistent with the scope of the invention.

CLAIMS:

1. A data processing system, comprising:

a cache system;

a cache miss controller coupled to the cache system;

a write-back controller coupled to the cache system;

a snoop controller coupled to the cache system;

means for a write-back buffer;

means for a reload buffer;

means for a snoop; and

a plurality of data pointers,

wherein each data pointer is configured to select cache entry for specific purpose;

reload pointer selection logic means capable of selecting a reload buffer line;

victim selection logic means capable of determining the stalest line, through skipping over data pointers; and

a snoop logic means for snooping bus operations for reacting to other devices' requests.

2. The system of claim 1, further comprising said cache miss controller coupled via said reload selection logic to the reload buffer line.

3. The system of claim 1, further comprising said write-back controller coupled via said victim selection logic to the victim buffer line.

4. The system of claim 1, further comprising said snoop controller coupled to a directory.

5. The system of claim 3, further comprising the write-back controller configured to reserve the reload buffer as an empty buffer.

6. The system of claim 3, further comprising said write-back controller configured to specify the least-recently-used data line as the victim line.

7. A method for managing write-back and reload operations in a cache system, employing pointers and moveable buffers, comprising:

receiving storage access instructions in a cache system, wherein each said instruction loads from a processor;

storing pointers in said cache system, wherein each pointer is pointing to one of the entry in cache storage;

executing a victim entry selection operation from victim entry calculation logic;

executing a reload entry selection operation from free entry calculation logic;

determining a victim entry in accordance with the processor's demand storage access history;

determining a reload entry in accordance with all cache states;

reserving a buffer as an exclusive reload buffer to place retrieved data for cache miss handling;

executing write-back to memory from victim entry freeing space for further cache miss handling.

8. The method of claim 7, further comprising a step of pointing a victim entry by variable pointer.

9. The method of claim 7, further comprising a step of variably pointing a reload entry.

10. The method of claim 7, further comprising a step of the victim selection logic determining when modified data needs writing back to memory, freeing an entry for further cache miss handling.

11. The method of claim 7, further comprising a step of the victim selection logic addressing and skipping over reload pointer to find the least-recently-used stale buffer.

12. A computer program product for authenticating code in a computer system, the computer program product having a medium with a computer program embodied thereon, the computer program comprising:

computer code for generating parallel write-back and reload instructions through next victim selection logic; and

computer code for ordering a data buffer movement command having an associative data buffer pointer movement command.

13. A cache system for providing data storage and processing in a computer system, including a computer program comprising:

computer code for generating parallel write-back and reload instructions through next victim selection logic; and

computer code for moving a write-back buffer with computer code for moving a reload buffer having an associative data pointer movement command.

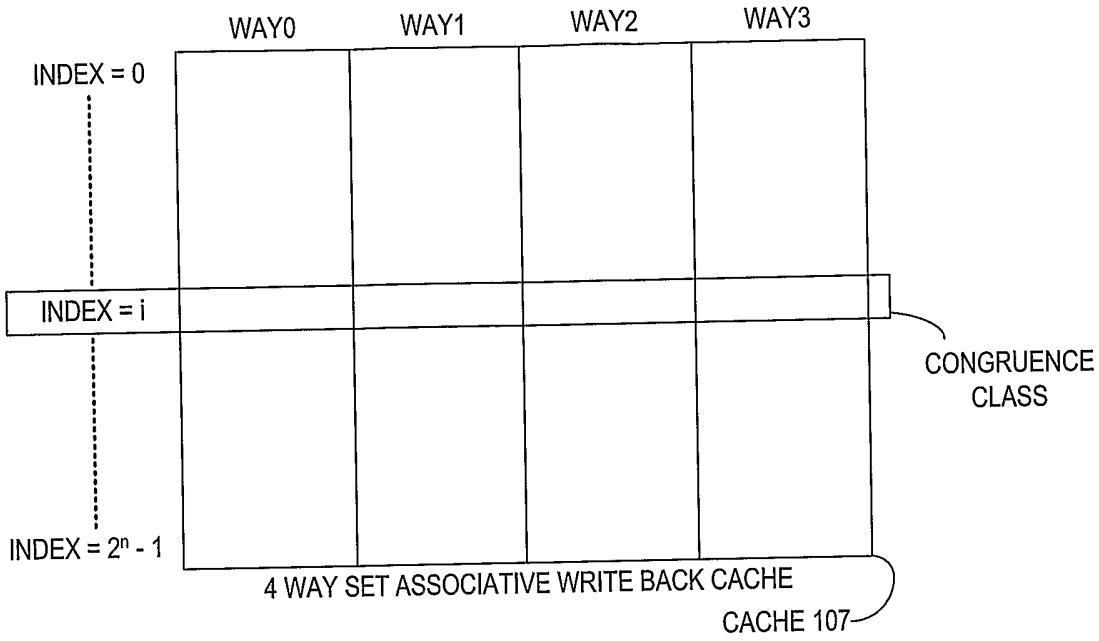

WAY0
WAY1
WAY2
WAY3
INDEX = 0
INDEX = i
INDEX = 2^n - 1
CONGRUENCE CLASS
4 WAY SET ASSOCIATIVE WRITE BACK CACHE
CACHE 107

FIG. 1A

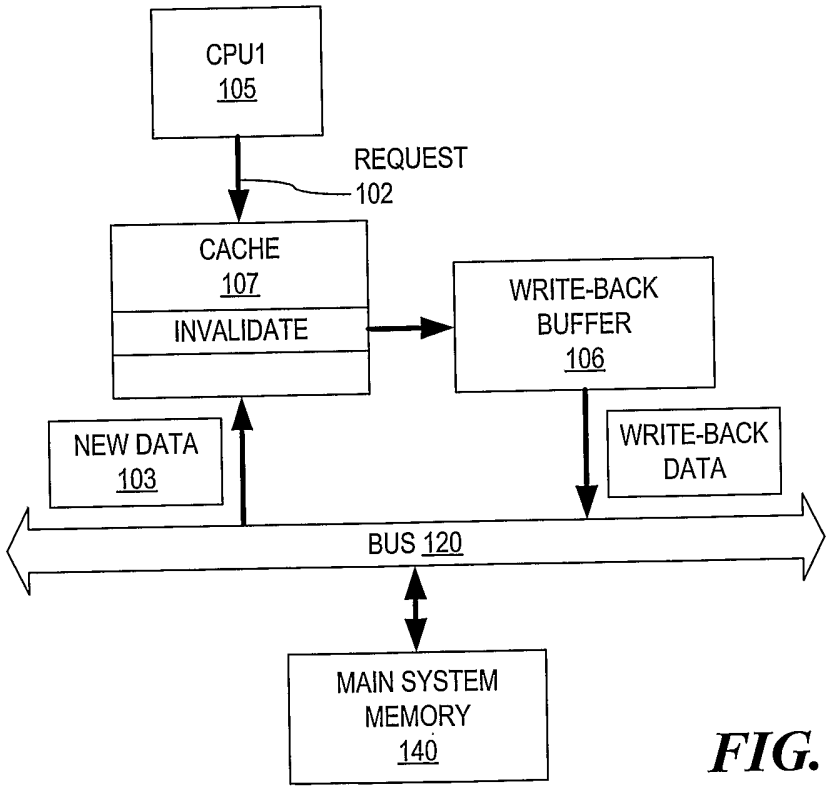

CPU1
105
REQUEST
102
CACHE
107
INVALIDATE
WRITE-BACK
BUFFER
106
NEW DATA
103
WRITE-BACK
DATA
BUS 120
MAIN SYSTEM
MEMORY
140

FIG. 1B

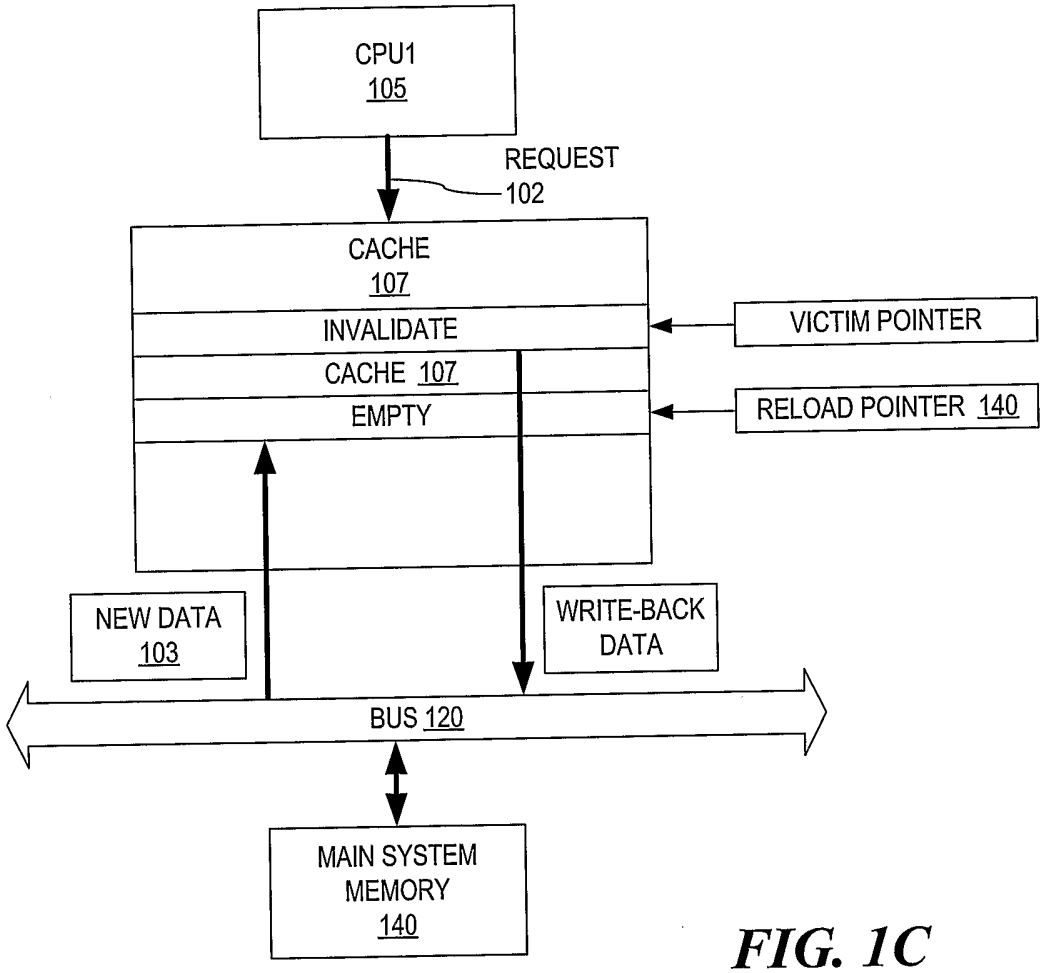

CPU1
105
REQUEST
102
CACHE
107
INVALIDATE
CACHE 107
EMPTY
VICTIM POINTER
RELOAD POINTER 140
NEW DATA
103
WRITE-BACK
DATA
BUS 120
MAIN SYSTEM
MEMORY
140

FIG. 1C

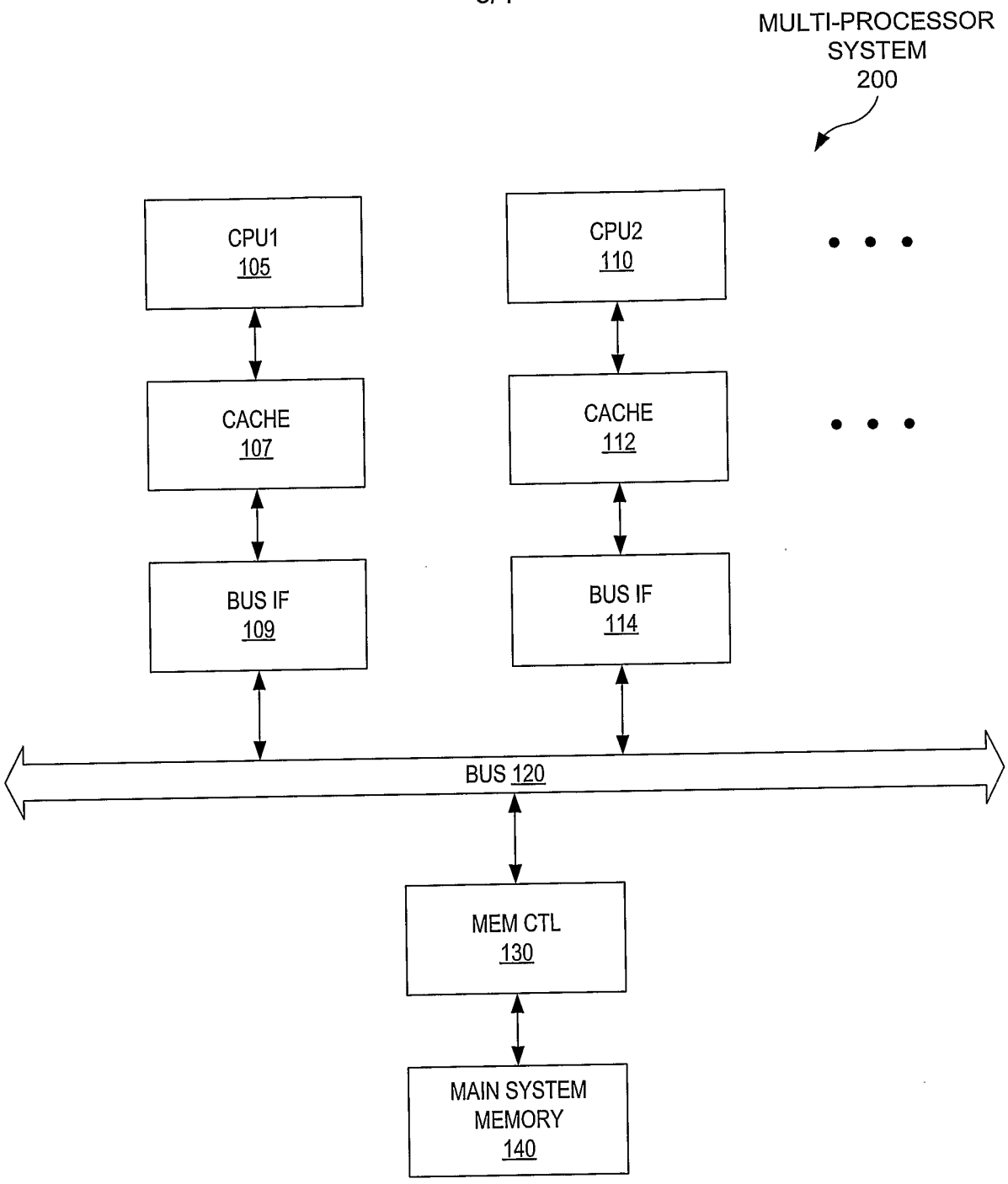

MULTI-PROCESSOR SYSTEM 200
CPU1 105
CPU2 110
CACHE 107
CACHE 112
BUS IF 109
BUS IF 114
BUS 120
MEM CTL 130
MAIN SYSTEM MEMORY 140

FIG. 2

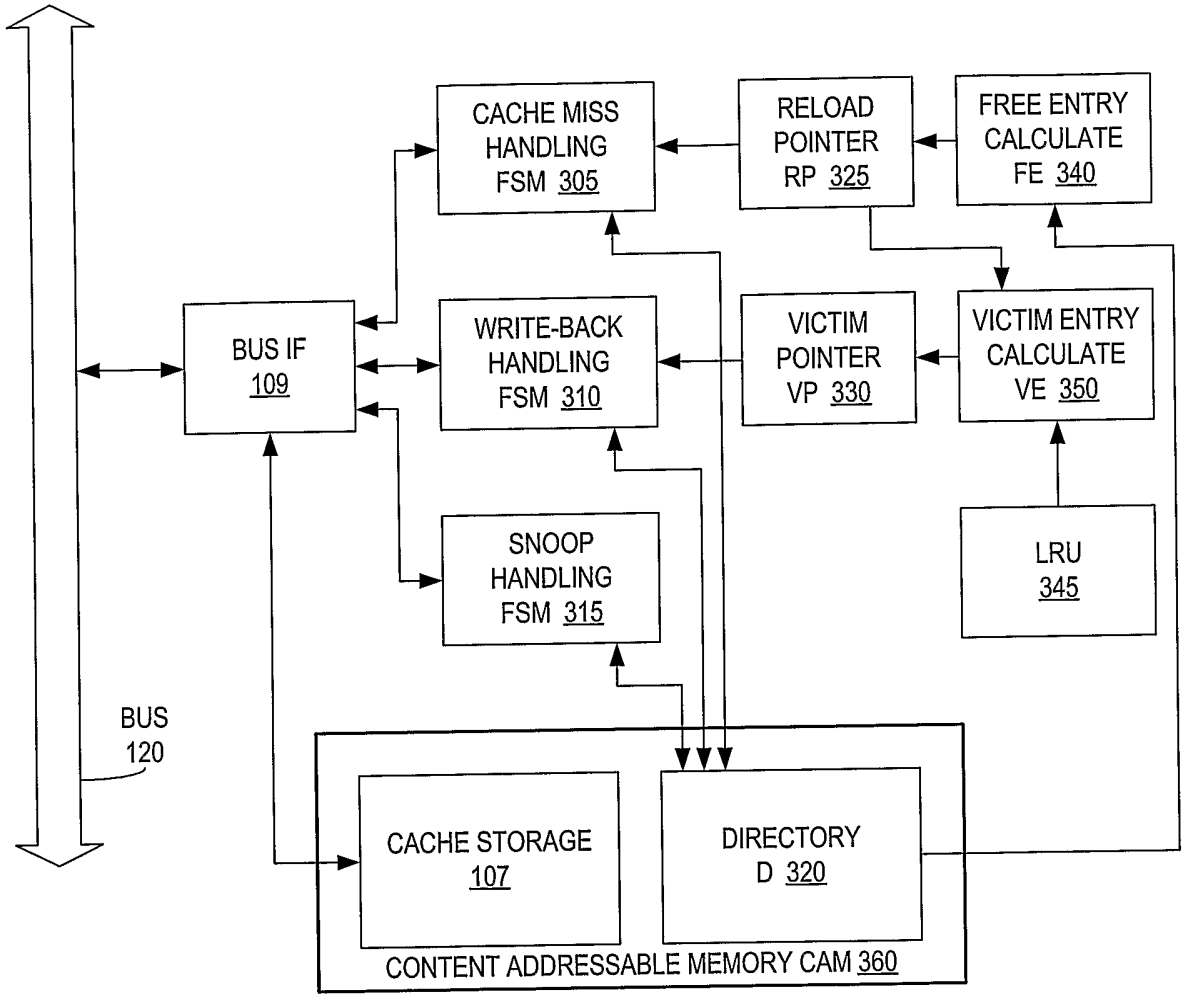

CACHE MISS HANDLING FSM 305
RELOAD POINTER RP 325
FREE ENTRY CALCULATE FE 340
BUS IF 109
WRITE-BACK HANDLING FSM 310
VICTIM POINTER VP 330
VICTIM ENTRY CALCULATE VE 350
SNOOP HANDLING FSM 315
LRU 345
BUS 120
CACHE STORAGE 107
DIRECTORY D 320
CONTENT ADDRESSABLE MEMORY CAM 360

FIG. 3

INTERNATIONAL SEARCH REPORT

International application No
PCT/IB2005/003565

A. CLASSIFICATION OF SUBJECT MATTER
G06F12/12 G06F12/08

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

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2003/159004 A1 (TSERN ELY K ET AL) 21 August 2003 (2003-08-21) figure 5 paragraph [0002] paragraph [0015] paragraph [0033] - paragraph [0037] paragraph [0072] - paragraph [0073] -----	1-13
X	US 5 950 205 A (AVIANI, JR. ET AL) 7 September 1999 (1999-09-07) the whole document -----	1-13

☐ 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 document 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 another 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
16 March 2006	23/03/2006

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Filip, L

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/IB2005/003565

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2003159004 A1	21-08-2003	US 2002040416 A1	04-04-2002
US 5950205 A	07-09-1999	NONE	