COMMONWEALTH OF AUSTRALIA

Patents Act 1952-1969

CONVENTION APPLICATION FOR A PATENT

(1) Here insert (in full) Name or Names of Applicant or Applicants, followed by Address (es).

~~We~~ XX (1) INTERNATIONAL COMPUTERS LIMITED, of ICL House, Putney, London. SW15 1SW. England.

(2) Here insert Title of Invention.

hereby apply for the grant of a Patent for an invention entitled: (2) MULTI-CACHE DATA STORAGE SYSTEM

(3) Here insert number(s) of basic application(s)

which is described in the accompanying complete specification. This application is a Convention application and is based on the application numbered (3) 8728494

(4) Here insert Name of basic Country or Countries, and basic date or dates

for a patent or similar protection made in (4) United Kingdom on 5th December, 1987.

APPLICATION ACCEPTED AND AMENDMENTS ALLOWED 29.1.91

~~My~~ Our address for service is Messrs. Edwd. Waters & Sons, Patent Attorneys, 50 Queen Street, Melbourne, Victoria, Australia.

DATED this 1st day of December. 19 88

(5) Signature (s) of Applicant (s) or Seal of Company and Signatures of its Officers as prescribed by its Articles of Association.

(5) INTERNATIONAL COMPUTERS LIMITED

By:

L.J. DYSON

Registered Patent Attorneys

02/12/88

M004855

To:

THE COMMISSIONER OF PATENTS.

Edwd. Waters & Sons, Melbourne.

(CONVENTION. Company.) Form 8

COMMONWEALTH OF AUSTRALIA

Patents Act 1952-1969

DECLARATION IN SUPPORT OF A CONVENTION APPLICATION FOR A PATENT OR PATENT OF ADDITION

(1) Here insert (in full) Name of Company.

In support of the Convention Application made by(1) International Computers Limited

(hereinafter referred to as the applicant) for a Patent

(2) Here insert title of Invention.

for an invention entitled:(2) MULTI-CACHE DATA STORAGE SYSTEM

(3) Here insert full Name and Address, of Company official authorized to make declaration.

I,(3) Roger Gaymer Broadie of STC Patents, West Road, Harlow, Essex. CM20 2SH.

do solemnly and sincerely declare as follows:

1. I am authorised by the applicant for the patent to make this declaration on its behalf.

2. The basic application as defined by Section 141 of the Act was made in(4) Great Britain on the Fifth day of December 19 87, by International Computers Limited

~~on the day of 19 by~~

(4) Here insert basic Country or Countries followed by date or dates and basic Applicant or Applicants.

(5) Here insert (in full) Name and Address of Actual Inventor or Inventors.

3.(5) David Paul Crane of 24, Hillbrow, Reading, Berkshire. RG2 8JD: Terence Michael Cole of 12, Rosecroft Way, Shinfield, Reading, Berkshire. RG2 9AP and Geoffrey Poslitt of 57, Moray Avenue, College Town Camberley, Surrey. GU15 4XE.

~~is~~/are the actual inventors of the invention and the facts upon which the applicant is entitled to make the application are as follow:

The applicant is the assignee of the said Inventors by virtue of of an Assignment dated 27th November 1987

4. The basic application referred to in paragraph 2 of this Declaration was the first application made in a Convention country in respect of the invention the subject of the application.

DECLARED at Harlow, Essex. England.

this Eighteenth day of October 19 88

(6) Signature.

(6) R G Broadie

Roger Gaymer Broadie
Authorised Signatory

To: THE COMMISSIONER OF PATENTS.

Edwd. Waters & Sons, Melbourne.

(12) PATENT ABRIDGMENT (11) Document No. AU-B-26541/88
(19) AUSTRALIAN PATENT OFFICE (10) Acceptance No. 609228

(54) Title
MULTI-CACHE DATA STORAGE SYSTEM

International Patent Classification(s)
(51)[5] G06F 012/08
(51)[4] G06F 013/00

(21) Application No. : 26541/88 (22) Application Date : 02.12.88

(30) Priority Data

(31) Number	(32) Date	(33) Country
8728494	05.12.87	GB UNITED KINGDOM

(43) Publication Date : 08.06.89

(44) Publication Date of Accepted Application : 26.04.91

(71) Applicant(s)
INTERNATIONAL COMPUTERS LIMITED

(72) Inventor(s)
DAVID PAUL CRANE; TERENCE MICHAEL COLE; GEOFFREY POSLITT

(74) Attorney or Agent
WATERMARK PATENT & TRADEMARK ATTORNEYS, Locked Bag 5, HAWTHORN VIC 3122

(56) Prior Art Documents
WO 90/00283
AU 42304/89 G06F 12/08 13/00
US 4785398

(57) Claim

1. A multi-processor data processing system comprising a plurality of data processing units, each data processing unit having an associated cache unit, the cache units being connected by a bus to a main memory, wherein each cache unit comprises:

(a) a data cache, addressed by a virtual address from the associated processing unit, to allow the processing unit to access data from the cache for reading or updating,

(b) means for translating the virtual address into a physical address; and for transmitting the physical address over the bus to the main memory whenever the cache is updated by its associated processing unit,

(c) means for continuously monitoring the bus to detect physical addresses transmitted from

.../2

other cache units, and

(d) means responsive to a physical address received over the bus, for determining whether a data item corresponding to that physical address is present in the data cache, and, if so, updating or invalidating that data item in the data cache.

609228

Form 10

COMMONWEALTH OF AUSTRALIA

PATENTS ACT 1952-69

COMPLETE SPECIFICATION

(ORIGINAL)

Class Int. Class

Application Number:
Lodged:

Complete Specification Lodged:
Accepted:
Published:

Priority :

Related Art :

This document contains the amendments made under Section 49 and is correct for printing.

Name of Applicant : INTERNATIONAL COMPUTERS LIMITED,

Address of Applicant : ICL House, Putney, London SW15 1SW, England.

Actual Inventor: DAVID PAUL CRANE, TERENCE MICHAEL COLE and GEOFFREY POSLITT

Address for Service : EDWD. WATERS & SONS,
50 QUEEN STREET, MELBOURNE, AUSTRALIA, 3000.

Complete Specification for the invention entitled:

MULTI-CACHE DATA STORAGE SYSTEM

The following statement is a full description of this invention, including the best method of performing it known to us :-

MULTI-CACHE DATA STORAGE SYSTEM.

This invention relates to multi-cache data storage systems. More specifically, the invention is concerned with a data storage system of the kind comprising a main store and a plurality of smaller, faster cache stores. The invention is particularly although not exclusively concerned with a storage system of this kind for use in a multi-processor data processing system, in which each of the cache stores is associated with a respective one of the processors, and the main store is shared among the processors.

In such a system, when one of the cache stores is updated, it is in general also necessary to update the corresponding data in the main store. If the same data is also held in any of the other cache stores, it will now be inconsistent with the data in the main store. This is referred to as the problem of cache coherency.

One way of overcoming this problem is to arrange for each cache to continuously monitor all updates to the main store from any other cache. If any of these updates relate to a data item in the cache, it is updated or invalidated.

In a adata storage system using virtual addressing, there are two possibilities for addressing the cache store: either with the virtual address or with the corresponding physical (real) address. Using the

virtual address has the advantage that it does not need to be translated into the physical address before the cache can be accessed, and hence cache access is faster.

In order to ensure cache co-herency in a multi-cache system with virtually addressed caches, it has been proposed that, when a cache updates a data item, it broadcasts the virtual address of that data item to all the other caches, as well as sending the physical address to the main store. Each cache can therefore monitor the virtual addresses from the other caches and use them to invalidate the corresponding data items so as to ensure cache coherency. However, this solution increases the number of connections required between the caches.

The object of the present invention is to avoid this need to send both the virtual address and the physical address.

Summary of the Invention

According to the invention, there is provided a multi-processor data processing system comprising a plurality of data processing units, each data processing unit having an associated cache unit, the cache units being connected by a bus to a main memory, wherein each cache unit comprises:

(a) a data cache, addressed by a virtual address from the associated processing unit, to allow the processing unit to access data from the cache for reading or updating,

(b) means for translating the virtual address into a physical address; and for transmitting the physical address over the bus to the main

memory whenever the cache is updated by its associated processing unit,

(c) means for continuously monitoring the bus to detect physical addresses transmitted from other cache units, and

(d) means responsive to a physical address received over the bus, for determining whether a data item corresponding to that physical address is present in the data cache, and, if so, updating or invalidating that data item in the data cache.

Brief Description of the Drawings.

One multi-processor system including a multi-cache memory system in accordance with the invention will now be described by way of example with reference to the accompanying drawings.

Figure 1 is an overall block diagram of the multi-processor system, including a plurality of cache store units.

Figure 2 shows one cache store unit in more detail.

Figure 3 shows a physical address tag unit forming part of the cache store unit.

Description of an Embodiment of the Invention.

Referring to Figure 1, the multi-processor system comprises a plurality of processing units 10, and a shared main memory 11. Each processing unit 10 has its own cache unit 12. The cache units 12 are all connected to the main memory 11 by way of a high speed private memory bus 13.

In operation, when a processing unit 10 requires to read or write a data item, it sends the virtual address VA of that item to its cache unit 12. If the required data item is present in the cache, it can be accessed immediately by the processing unit, for reading or writing. Whenever one of the processors 10 updates a data item in its cache unit 12, the cache generates an update message over the bus to the main memory 11. This ensures that the main memory is kept consistent with the cache.

Each cache unit continuously monitors the bus 13 for any update messages from the the other cache units. Whenever a cache detects such a message, it checks whether the updated data item is present in the cache and, if so, invalidates it. This ensures cache coherency. The way in which this is done will be described below.

Referring now to Figure 2, this shows one of the cache units 12 in detail.

The cache comprises a data random-access memory (RAM) 20, holding 128K bytes. The RAM is organised as 4K individually addressable lines, each line holding 32 bytes. The data RAM is addressed by bits VA 5-16 of a 32-bit virtual address VA 0-31, so as to access one line of data for reading or writing. Bits VA 0-4 specify one byte within the address line, but are not used within the cache.

The cache also includes a VA tag RAM 21, which holds 4K tags, one for each line of data in the data RAM. Each tag represents bits VA 17-31 of the virtual address of the data held in this line. The VA tag RAM is addressed by bits VA 5-16.

The cache further includes a status tag RAM 22, which holds 4K status tags, one for each line of the data RAM. Each of these tags includes a validity bit indicating whether or not the corresponding line of data is valid. The status tag RAM is also addressed by bits VA 5-16.

Whenever the processing unit 10 requires to access a data item, it sends the virtual address VA 5-31

to the cache. Bits VA 5-16 access one line of the cache, and its corresponding VA tag and status tag. The VA tag from the RAM 21 is then compared with bits VA 17-31 of the virtual address from the processor, by means of a comparator 23. The result is then combined in an AND gate 24 with the validity bit for the RAM 22, so as to produce a HIT signal.

Thus, HIT is true only if the VA tag of the addressed line of data matches the corresponding bits of the virtual address, and that line is valid. The HIT signal therefore indicates to the processing unit that the required data is present in the cache, and can be read or updated as required.

The cache unit also includes a memory management unit (MMU) 25, which translates the virtual address bits VA 5-31 into a physical address PA 5-31. The address space is divided into pages of 8K bytes. Bits VA 13-31 constitute a virtual page address, and are translated by the MMU into a corresponding physical page address PA 13-31. Bits VA 5-12 specify a line of 32 bytes within the page, and are used directly as the corresponding bits PA 5-12 of the physical address, without any translation.

Whenever the processor updates a line of data in the cache, the cache sends an update message over the bus 13 to the main memory. This update message includes the physical address bits PA 5-31, and also the updated value of the data line.

The cache unit continuously monitors the bus 13 for update messages from other cache units. Whenever it detects such a message, it captures the physical address bits PA 5-31 of the message in a register 26.

The PA from the register 26 is applied to a PAtag logic circuit 27. As will be described in detail below, the PA tag logic holds as tags the physical addresses of all the lines of data currently in the cache. The PA tag logic effectively performs a parallel search on all the PA tags held in it, and this determines whether or not the corresponding line of data is held in the cache. If a match is detected, the PA tag logic outputs the virtual address bits VA 5-16, indicating the position of the data line in the data RAM 20. These VA bits are then used to address the status tag RAM 22, and the validity bit of the addressed status tag is cleared. This invalidates the corresponding data line and hence ensures cache coherency.

It can be seen that the PA tag logic 27 must effectively perform a comparison between the PA from the bus and the PA tags of all 4K lines in the cache. At first sight, this requires a large associative memory, capable of performing 4K comparisons in parallel. However, as will be shown, the present embodiment avoids the need for such a large number of simultaneous comparisons.

Referring now to Figure 3 this shows the PA tag logic in detail.

This logic comprises sixteen PA tag RAMs 30, each of which holds 256 tags. There are thus 256 x 16 = 4K tags altogether, one for each of the data lines in the data RAM.

All the RAMs 30 are addressed in parallel by the outputs of a multiplexer 31, which selects either bits PA 5-12 from the bus (by way of register 26), or bits VA 5-12 from the processing unit.

Any one of the RAMs 30 can be selected by meansof bits VA 13-16 from the processing unit. These bits are decoded by a decoder 32 to produce a write enable signal for the selected one of the RAMs.

All the tag RAMs 30 receive a common input signal from a multiplexer 33, which selects either bits PA 13-31 from the MMU 25, or bits PA 13-31 from the bus.

Whenever a new line of data is loaded into the cache, the PA tag RAMs are addressed by VA 5-12 and VA 13-16, so as to select one tag in one of the RAMs. The physical address bits PA 13-31 from the MMU are then written into the selected tag, as a PA tag for the new line of data. In this way, the PA tag logic keeps tags indicating the physical addresses of all the data lines in the cache.

Whenever an update message is detected in the bus, the tag RAMs 30 are all addressed by PA 5-12 from the bus. The bits PA 13-31 from the bus are then compared, in parallel, with the addressed tag in each of the tag RAMs, i.e. sixteen comparisons are performed in parallel, one by each of the tag RAMs. Each of the tag RAMs has a HIT output line, which indicates a match between the input PA 13-31 and the stored tag.

For reasons to be explained below, only one (or none) of the HIT outputs will be true at any given time. The HIT outputs are applied to a 16:4 encoder 34, which encodes the position of the true HIT signal. The output of the encoder 34 thus reconstructs bits VA 13-16 of the virtual address of the data line with physical address equal to bits PA 5-21 from the bus. The reconstructed VA 13-16 from the encoder are combined with bits PA 5-12 from the bus, to provide the required virtual address VA 5-16 for invalidating the status tag RAM, as described above.

If, on the other hand, none of the HIT signals from the tag RAMS 30 are true, then the encoder 34 produces a MISS signal, indicating that the data line in question is not present in the cache.

Thus, it can be seen that the PA tag logic shown in Figure 3 can check whether a data line with a particular PA tag is present in the cache, by performing only sixteen comparisons in parallel, rather than having to perform 4K comparisons. This is achieved by making use of the fact that, of the bits VA 5-16 which are used to address the cache RAMs, bits VA 5-12 are identical with bits PA 5-12 of the physical address; only bits VA 13-16 are different from the physical address. Hence, when a physical address is received from the bus, there are only 16 possible lines in the cache which might hold data with this PA.

The PA tag RAMs 30 may be implemented, for example, by meas of four Fujitsu MB 81 C50 CMOS TAG Random Access Memories. Each of these devices provides a four-way tag RAM, including comparators for performing the tag comparisons, and can therefore be used to implement a group of four of the tags RAMs 30.

In a virtual address memory system, several different virtual addresses may map on to the same physical address, these being referred to as synonyms. In general, this means that there may be more than one line in the cache corresponding to a particular physical address. In the present system, this is avoided by a software restriction, which requires that synonyms must be aligned on some multiple of 128K byte boundaries (i.e the size of the cache). This means that synonyms will all map on to he same line of the cache, and hence, at any given time, only one of them can be present in the

cache. This is the reason why only one of the tag RAMs 30 can produce a HIT at any given time.

In the above description, it was assumed that a "write through" strategy is adopted for updates, i.e. that the main store is updated whenever the cache is updated. However, it will be appreciated that the invention is equally applicable to storage systems using a "copy back" strategy, i.e. in which data is updated in the main store only when it is about to be overwritten in the cache and has been modified since it was loaded into the cache.

Also in the above description, when a hit is detected by the PA tag logic, the corresponding data item is invalidated in the cache. Alternatively, the data item may be updated with the new value from the bus, which will also ensure cache coherency.

It should be noted that the virtual address VA 0-31 may be extended in known manner by a context number to create a full virtual address.

THE CLAIMS DEFINING THE INVENTION ARE AS FOLLOWS:

1. A multi-processor data processing system comprising a plurality of data processing units, each data processing unit having an associated cache unit, the cache units being connected by a bus to a main memory, wherein each cache unit comprises:

(a) a data cache, addressed by a virtual address from the associated processing unit, to allow the processing unit to access data from the cache for reading or updating,

(b) means for translating the virtual address into a physical address; and for transmitting the physical address over the bus to the main memory whenever the cache is updated by its associated processing unit,

(c) means for continuously monitoring the bus to detect physical addresses transmitted from other cache units, and

(d) means responsive to a physical address received over the bus, for determining whether a data item corresponding to that physical address is present in the data cache, and, if so, updating or invalidating that data item in the data cache.

2. A system according to claim 1 wherein the means for determining whether a data item is present in the data cache comprises

(a) a tag memory for holding a physical address tag for each data item in the data cache,

(b) means for comparing the physical address received over the bus with the physical address tags in the tag memory and, if a match is detected, producing an invalidation address equal to the virtual address of the data item having that physical address, and

(c) means for using the invalidation address to access the data cache to invalidate the corresponding data item in the data cache.

3. A system according to claim 2 wherein

(a) the virtual address comprises first, second and third portions, the first portion being used directly to provide a first portion of the physical address without translation and the second and third portions being translated to provide a second portion of the physical address, the first and second portions of the virtual address being used to address the data cache,

(b) when a data item is loaded into the data cache, the first and second portions of the virtual address are used to access the tag memory and the second portion of the physical address is written into the tag memory as a physical address tag;

(c) when a physical address is received over the bus, the first portion of the physical address is used to select a set of tags in the tag memory and these tags are compared in parallel with the second portion of the physical address,

(d) the result of the comparison is encoded and combined with the first portion of the physical address received over the bus, to form said invalidation address.

4. A system according the claim 3 wherein the tag memory comprises a plurality of sections each of which holds a plurality of tags, and wherein, when data is loaded into the data cache, the first portion of the

virtual address of that item is used to address all the sections of the tag memory in parallel, and the second portion of the virtual address is used to select one of the sections.

5. A system according to claim 4 wherein, when a physical address is received over the bus, the first portion of that address is used to address all the sections of the tag memory in parallel, and the second portion of the physical address is compared with all the portions in parallel.

6. A multi-processor data processing system substantially as hereinbefore described with reference to the accompanying drawings.

DATED this 4th day of January, 1991.

INTERNATIONAL COMPUTERS LIMITED

WATERMARK PATENT &
TRADE MARK ATTORNEYS
"THE ATRIUM"
2ND FLOOR, 290 BURWOOD ROAD
HAWTHORN VIC. 3122
AUSTRALIA.

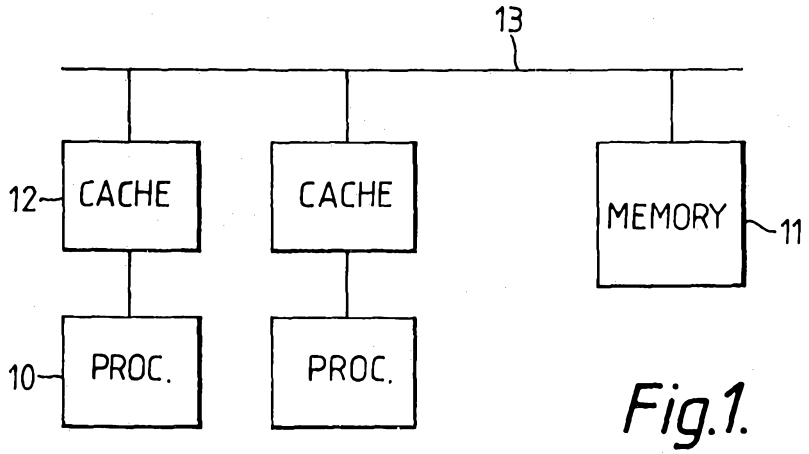
13
12
CACHE
CACHE
MEMORY
11
10
PROC.
PROC.

Fig.1.

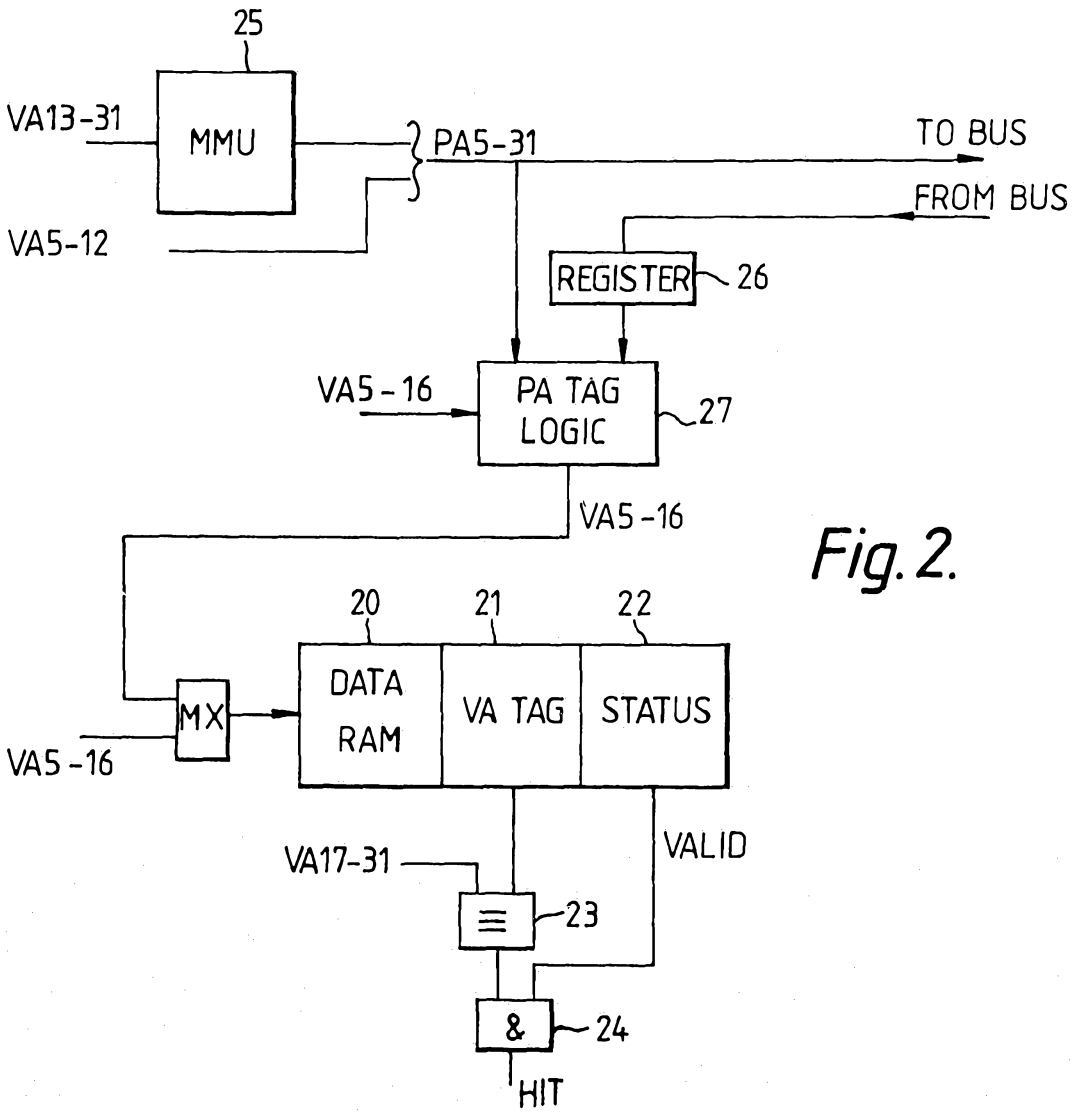
25
VA13-31
MMU
PA5-31
TO BUS
FROM BUS
VA5-12
REGISTER
26
VA5-16
PA TAG LOGIC
27
VA5-16
20
21
22
MX
DATA RAM
VA TAG
STATUS
VA5-16
VA17-31
VALID
23
&
24
HIT

Fig. 2.

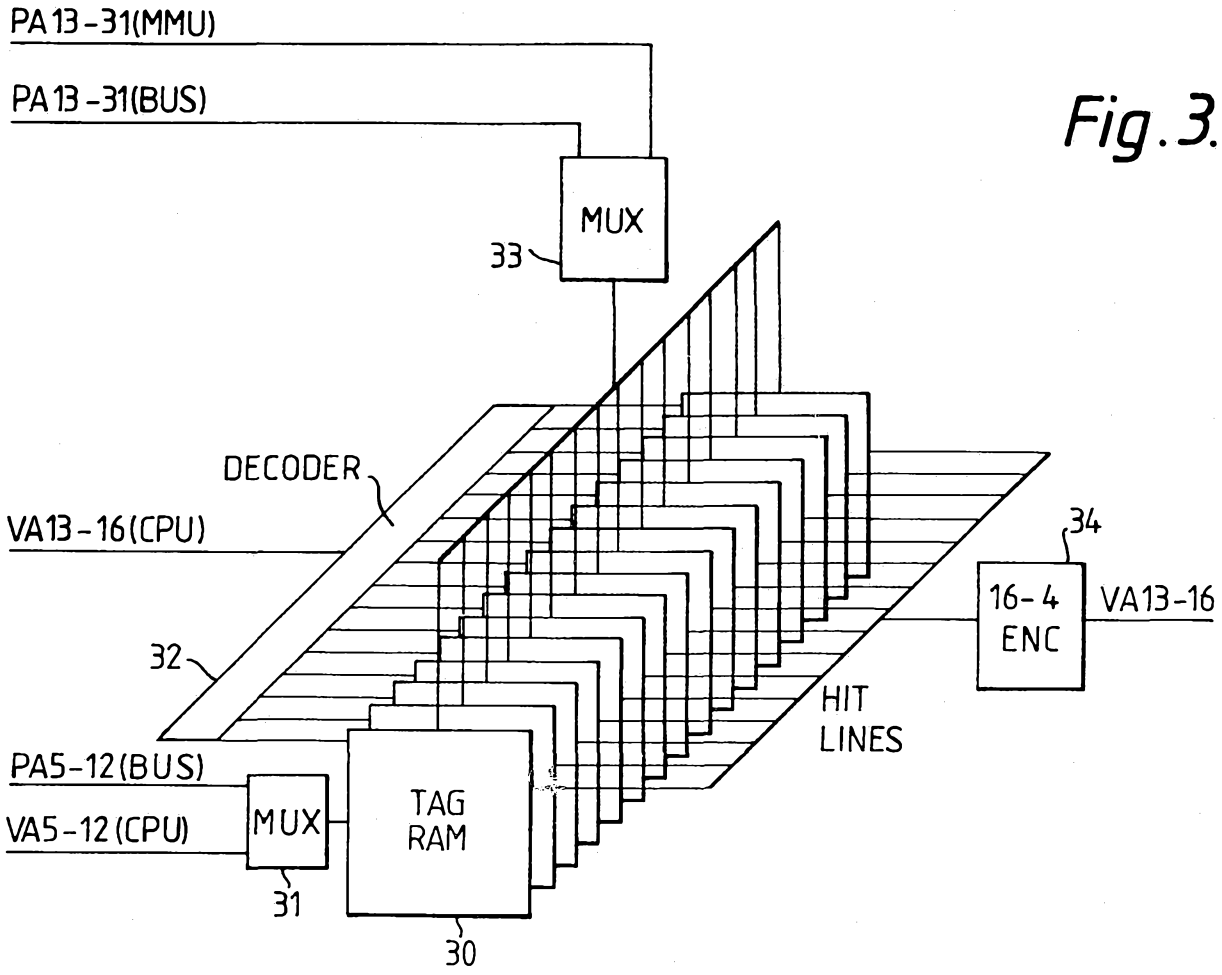
PA13-31(MMU)
PA13-31(BUS)
MUX
33
DECODER
VA13-16(CPU)
32
34
16-4
ENC
VA13-16
HIT
LINES
PA5-12(BUS)
VA5-12(CPU)
MUX
31
TAG
RAM
30

Fig.3.