PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G01R 31/28, G06F 11/00 | A1

(11) International Publication Number: **WO 97/24623**

(43) International Publication Date: 10 July 1997 (10.07.97)

(21) International Application Number: PCT/US96/11840

(22) International Filing Date: 17 July 1996 (17.07.96)

(30) Priority Data:
08/579,414 27 December 1995 (27.12.95) US

(71) Applicant *(for all designated States except US)*: INTEL CORPORATION [US/US]; 2200 Mission College Boulevard, Santa Clara, CA 95052 (US).

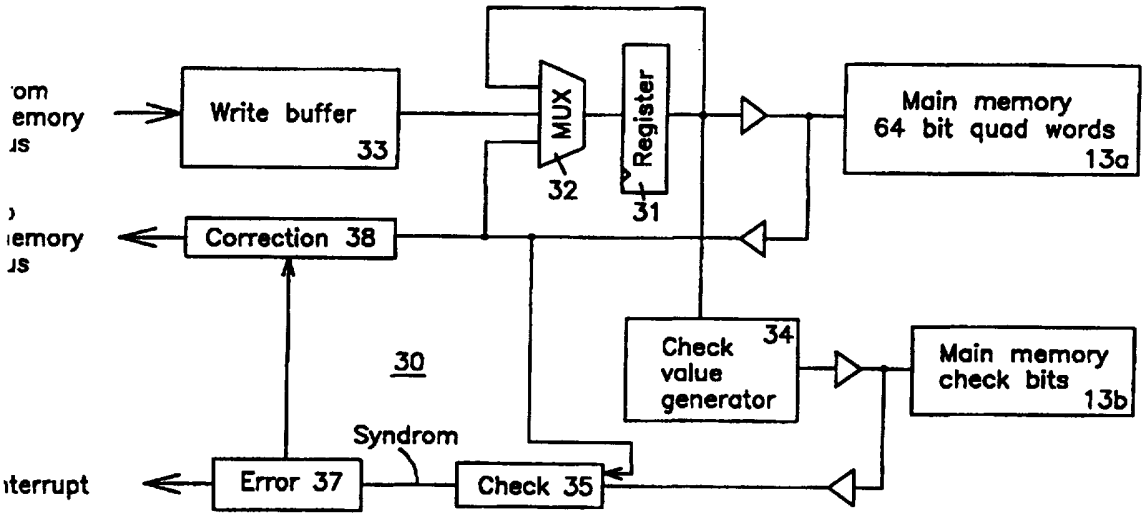
(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: LANGENDORF, Brian, K. [US/US]; 1910 Linhurst Court, El Dorado Hills, CA 95762 (US). DERR, Michael [US/US]; 109 Gable Street, Folsom, CA 95762 (US).

(74) Agents: KING, Stephen, L. et al.; Blakely, Sokoloff, Taylor & Zafman, 7th floor, 12400 Wilshire Boulevard, Los Angeles, CA 90025 (US).

(81) Designated States: AL, AM, AT, AT (Utility model), AU, AZ, BB, BG, BR, BY, CA, CH, CN, CU, CZ, CZ (Utility model), DE, DE (Utility model), DK, DK (Utility model), EE, EE (Utility model), ES, FI, FI (Utility model), GB, GE, HU, IL, IS, JP, KE, KG, KP, KR, KZ, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SK (Utility model), TJ, TM, TR, TT, UA, UG, US, UZ, VN, ARIPO patent (KE, LS, MW, SD, SZ, UG), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

Published
With international search report.

(54) Title: METHOD AND APPARATUS FOR COMBINING WRITES TO MEMORY



(57) Abstract

Apparatus and a method for testing each write to memory to determine whether it is addressed to an address identical to that of another write to memory (13a) waiting to be processed and merging the valid data in any subsequent writes to the same address until a memory write occurs.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AM	Armenia	**GB**	United Kingdom	**MW**	Malawi
AT	Austria	**GE**	Georgia	**MX**	Mexico
AU	Australia	**GN**	Guinea	**NE**	Niger
BB	Barbados	**GR**	Greece	**NL**	Netherlands
BE	Belgium	**HU**	Hungary	**NO**	Norway
BF	Burkina Faso	**IE**	Ireland	**NZ**	New Zealand
BG	Bulgaria	**IT**	Italy	**PL**	Poland
BJ	Benin	**JP**	Japan	**PT**	Portugal
BR	Brazil	**KE**	Kenya	**RO**	Romania
BY	Belarus	**KG**	Kyrgystan	**RU**	Russian Federation
CA	Canada	**KP**	Democratic People's Republic of Korea	**SD**	Sudan
CF	Central African Republic			**SE**	Sweden
CG	Congo	**KR**	Republic of Korea	**SG**	Singapore
CH	Switzerland	**KZ**	Kazakhstan	**SI**	Slovenia
CI	Côte d'Ivoire	**LI**	Liechtenstein	**SK**	Slovakia
CM	Cameroon	**LK**	Sri Lanka	**SN**	Senegal
CN	China	**LR**	Liberia	**SZ**	Swaziland
CS	Czechoslovakia	**LT**	Lithuania	**TD**	Chad
CZ	Czech Republic	**LU**	Luxembourg	**TG**	Togo
DE	Germany	**LV**	Latvia	**TJ**	Tajikistan
DK	Denmark	**MC**	Monaco	**TT**	Trinidad and Tobago
EE	Estonia	**MD**	Republic of Moldova	**UA**	Ukraine
ES	Spain	**MG**	Madagascar	**UG**	Uganda
FI	Finland	**ML**	Mali	**US**	United States of America
FR	France	**MN**	Mongolia	**UZ**	Uzbekistan
GA	Gabon	**MR**	Mauritania	**VN**	Viet Nam

METHOD AND APPARATUS FOR COMBINING WRITES TO MEMORY

BACKGROUND OF THE INVENTION

Field Of The Invention

This invention relates to personal computer systems and, more particularly, to methods and apparatus for merging different increments of data being written to the same address in memory so that write operations require fewer steps and may be accomplished more rapidly.

History Of The Prior Art

In order to increase the speed of operation of personal computer systems, the operational speed of central processing units has constantly been increased. In general, state of the art processors run faster than transfers take place in other parts of a personal computer system. Writes from the central processing unit to main memory are often quite slow. Because of this, many prior art systems utilize write buffers on the memory bus between the central processing unit and main memory. Such a write buffer stores data which a processor is writing to main memory until main memory is ready to accept it. Write buffers are particularly advantageous when used with processors and other circuitry which have been designed to transfer data in bursts, the individual elements of each burst including an increment of individually addressed data of a size up to the memory bus width. Burst transfers are fast because a processor transfers a number of increments of data to be written on a memory bus or a local bus during a single bus access using only a single set of control signals. Transferring a plurality of increments of data to a write buffer allows the processor to pursue other operations utilizing data and instructions in its caches without having to wait for the slower memory operations to complete.

However, when a processor functions faster than data can be written to memory for some significant period, there will ultimately be insufficient space in the write buffers to hold all of the different increments of data

being written. This is especially true in systems which store error correction code (ECC) with data in memory to enhance reliability because such systems write data to memory substantially more slowly than do systems which do not use error correction code.

Error correction code is generated as a part of an error correction process and is used to detect storage errors in memory arrays and correct some of those errors. An error correction process uses a math function to compute during data storage an error correction code (referred to herein as a check value or ECC value) which is unique to the data stored. The check value is stored in memory in association with the data. When the data is later read from memory, a determination is made whether the data read would produce the check value stored with the data. If the data would not produce the check value stored, some change has occurred in the data or the check value since they were first stored; and the data is untrustworthy. If the value has changed, then the data and the check value read from memory are sometimes used to accomplish the correction of the data depending on the type of error.

The amount of memory which is allotted to an ECC value is always a compromise, the more space allotted, the more accurate the check may be made but the less space remains for data. A typical ECC value for a sixty-four bit bus width increment of data stored in main memory may be eight bits. Such a value allows detection of all one and two bit errors, the detection of errors in four consecutive bits in certain types of memory, and the correction of all single bit errors. Computer system designers are usually unwilling to allot more than this percentage of space to memory reliability. Consequently, systems which use error correction code typically do not compute an ECC value for each byte, word, or doubleword stored individually, but only for an entire bus width increment of data. However, since writing any data of any length to memory changes the data from which the ECC value is computed, a new ECC value must be computed on a bus width increment of data each time data of any length is written to memory.

Writes to memory in bus width increments have little effect on the speed of writing. However, to accomplish a write to memory of data less than the width of the memory bus (e.g., one byte or one word of a double word wide bus), a memory controller typically must use a full bus width of data with which to compute an ECC value. To use a bus width of data for a memory write less than bus width, the memory controller must read the data at the address in memory, check the data against the ECC value stored to make sure the data is correct, modify the data read from memory with the new data being written, generate a new ECC value, and write the modified data back to memory in a bus width increment along with the new ECC value. This is a time consuming process.

For example, in a system using error correction code, if a processor writes four separate data increments to memory each of which is one word in length and the four writes are individually addressed to the sequential four word addresses of the same quad word space in memory so that each write to memory occurs separately, then each write requires the time consuming read/check ECC/modify/generate ECC/write process. On the other hand, if the same four words are written to the same memory space as a single quad word, only a single write operation is required. This operation writes an entire valid quad word which replaces the quad word in memory so the write includes no read or modify steps and only a single ECC value generation.

The total time required to transfer the data as four individual words is at least eight times as long as that required to write a single quad word. Even four quad words are written much more rapidly than are four individual words with a quad word memory bus because each quad word requires only a single write operation. Thus, writing less than bus width data increments in a system using error correction code slows the draining of the write buffer and ultimately slows the operation of the computer.

It is desirable to increase the speed of writing to memory in a personal computer system using a write buffer and practicing burst write operations on data stored with error correction code.

Summary Of The Invention

It is, therefore, an object of the present invention to provide apparatus and a method for increasing the speed at which data is written from a write buffer to a memory array in a personal computer system.

This and other objects of the present invention are realized in a personal computer including a write buffer having a plurality of storage areas each capable of storing all of the valid data included in a bus width write from a source along with the address and status of that data. The valid data stored at one of the storage areas is supplemented by any new valid data included in later writes to the same memory address until a memory controller is ready to write the data in that storage area to memory. When the memory controller is ready to write data at a storage area to memory, it tests that data to determine whether all of the data is valid. If the data is all valid, it is written to the addressed memory space. If some of the data in a storage area of the write buffer is invalid, the memory space to which the data is addressed is read and valid data stored in that memory space is merged with the valid data being written to provide a complete bus width of valid data to the addressed memory space. Merging valid data from succeeding writes in the write buffer reduces the number of less than full bus width writes requiring read/modify/write operations and associated ECC value tests and computations thereby speeding the operation of the computer system.

These and other objects and features of the invention will be better understood by reference to the detailed description which follows taken together with the drawings in which like elements are referred to by like designations throughout the several views.

Brief Description Of The Drawings

Figure 1 is a block diagram of a personal computer system designed in accordance with the present invention.

Figure 2 is a block diagram of a bridge circuit designed in accordance with the present invention included in the system of Figure 1.

Figure 3 is block diagram illustrating in more detail a write buffer which is a portion of the bridge circuit of Figure 2.

Figure 4 is a diagram illustrating storage of data in one embodiment of the write buffer illustrated in Figure 3.

Figure 5 is a block diagram illustrating in detail the implementation of one embodiment of the invention to provide the storage illustrated in Figure 4.

Notation And Nomenclature

Some portions of the detailed descriptions which follow are presented in terms of symbolic representations of operations on data bits within a computer memory. These descriptions and representations are the means used by those skilled in the data processing arts to most effectively convey the substance of their work to others skilled in the art. The operations are those requiring physical manipulations of physical quantities. Usually, though not necessarily, these quantities take the form of electrical or magnetic signals capable of being stored, transferred, combined, compared, and otherwise manipulated. It has proven convenient at times, principally for reasons of common usage, to refer to these signals as bits, values, elements, symbols, characters, terms, numbers, or the like. It should be borne in mind, however, that all of these and similar terms are to be associated with the appropriate physical quantities and are merely convenient labels applied to these quantities.

Further, the manipulations performed are often referred to in terms, such as adding or comparing, which are commonly associated with mental

operations performed by a human operator. No such capability of a human operator is necessary or desirable in most cases in any of the operations described herein which form part of the present invention; the operations are machine operations. Useful machines for performing the operations of the present invention include general purpose digital computers or other similar devices. In all cases the distinction between the method operations in operating a computer and the method of computation itself should be borne in mind. The present invention relates to a method and apparatus for operating a computer in processing electrical or other (e.g. mechanical, chemical) physical signals to generate other desired physical signals.

Detailed Description

Referring now to Figure 1, there is illustrated a computer system 10 configured in accordance with one embodiment of the present invention. The system 10 illustrated includes a central processing unit 11 which executes the various instructions provided to control the operations of the system 10. The central processing unit 11 is typically joined by a processor bus to a bridge circuit 14 which controls access to a local bus 12 adapted to carry information between the various components of the system 10. In Figure 1, the bus 12 is preferably a peripheral component interface (PCI) bus or other local bus adapted to provide especially fast transfers of data. In a typical system 10, various input/output devices are connected to the bus 12 as bus master and bus slave circuits. In the present illustration, for example, long term memory 15 may be joined to the PCI bus 12 as a bus slave circuit. Other input/output devices such as sound boards, frame buffers, and the like may also be joined to the bus 12.

The bridge circuit 14 is also joined by a memory bus to main memory 13. Main memory 13 is typically constructed of dynamic random access memory (DRAM) arranged in a manner well known to those skilled in the prior art to store information during a period in which power is provided to the system 10. Depending on the particular configuration,

the bridge circuit 14 will typically include a secondary cache controller, a secondary cache (often referred to as a L_2 cache), a memory controller, and a number of buffers for storing data during its transfer between the processor, main memory, and the local bus. These components may be an integrated part of the bridge circuit 14 or, alternatively, be joined on a circuit board to the other components.

Figure 2 illustrates in more detail the arrangement of major components of the bridge circuit 14 related to the present invention and main memory 13. As may be seen in Figure 2, the bridge circuit 14 includes a host interface 20 which joins the central processing unit 11 to the bridge circuit. The host interface 20 sends to and receives control signals from the central processing unit; it also sends to and receives control signals from various other components of the bridge circuit 14 including a host-to-main memory buffer controller 21 and a host-to-bus buffer controller 22. A bus arbiter circuit 23, typically not a part of the bridge circuit 14, controls access to the local bus 12. The host-to-main memory buffer controller 21, the host-to-bus buffer controller 22, and the bus arbiter circuit 23 all transfer control signals to a circuit 28 which generates control signals back to the bus arbiter circuit 23 to signal the state of operations related to bus access within the bridge circuit 14.

Also a part of the bridge circuit 14 are host-to-main memory buffers 25 and host-to-bus buffers 26. The host-to-main memory buffers 25 and host-to-bus buffers 26 respond to control signals from the host-to-main memory buffer controller 21 and the host-to-bus buffer controller 22 in carrying out their operations. The host-to-main memory buffers 25 include write buffers for buffering the transfer of write data provided from the central processing unit 11 (or, more generally, from any processor in other embodiments of the invention) on a host data bus for transfer on the memory bus to main memory 13. The host-to-bus buffer controller 22 controls the operation of host-to-bus buffers 27 which include write buffers for buffering the transfer of write data between the central processing unit 11 and the local bus 12 on the host data bus.

Figure 3 illustrates a circuit 30 in accordance with the present invention which is a portion of the host-to-main memory buffers 25 shown in Figure 2. The circuit 30 includes a write buffer 33 which receives the individual increments of data written to memory 13 (typically from the processor). In a system with a quad word memory bus, the write buffer 33 includes a plurality of individual storage areas (typically hardware registers) each divided into a portion capable of storing an increment of data of a size up to the bus width, a portion storing a memory address for the data, and possibly a portion storing a number of status bits. Figure 4 illustrates one embodiment of a write buffer 33 which includes eight individual storage areas each providing eight bytes of storage which may be allotted to the data included in eight sequential bus width write entries. As may be seen, each storage area in the buffer 33 of Figure 4 also includes storage for an address and status bits. Other write buffers may embody a lesser or greater number of individual storage areas each capable of storing more or less write data. Thus, another particular write buffer may furnish data storage for only four increments of bus width write data so that four sequential data writes furnished in a burst may be stored therein.

When data is first written to the write buffer 33, one of the plurality of storage areas is allotted; and the data is stored with its address pointing to that area. Although Figure 4 shows the address and status bits included in the same physical area of the buffer with each increment of data, the address and status bits might actually be stored apart from the data so long as they are associated with that area in a manner that the data in the area may be detected when data is written to a specific address by the central processing unit. Presuming that the write includes sufficient valid data to fill the storage area, all of the bytes are shown valid by the associated status bits. Succeeding writes to different addresses are each allocated an individual storage area and stored with associated address and status bits in a like manner.

Writes to memory from the buffer 33 occur in the sequence in which storage areas in the buffer 33 are allocated to write data for particular

memory addresses. The data in a storage area holding the oldest valid data in the write buffer 33 is selected as the next to be written when the host-to-main memory control circuit 21 ("the write controller 21") is ready to write a next increment of data to main memory 13. The controller 21 tests the validity of all of the bytes which are in that storage area of the buffer 33. Presuming that data to be written to memory is an entire quad word of valid data, the controller 21 enables a multiplexor 32 to transfer the entire quad word, the quad word address, and status bits to a register 31. When the data in the write buffer 33 is written to the register 31, the data is invalidated in the write buffer 33.

Since all bytes of the data are presumed valid, the controller 21 generates an enabling signal which causes the data in the register 31 to be written to the addressed space in the main memory 13. In Figure 3, the data is shown being written to a data storage portion 13a of main memory 13. At the same time, the data in the register 31 is also sent to a check value generator 34 (which may be, for example, a series of gates arranged to accept the data in parallel). The generator 34 generates and writes an ECC check value unique to the data to a check value portion 13b of the memory 13. The check value portion 13b may be physically with or apart from that memory which stores the data.

When the data in memory 13a is later read, the check value is also read. The data is transferred to a correction circuit 38 and to a check circuit 35. The check value is transferred to the check circuit 35 where the data and the check value cause a value referred to as a syndrom to be generated by the check circuit 35. The syndrom value is transferred to an error circuit 37 (which may be a lookup table or other correction arrangement known to those skilled in the art). If the data value read is correct, the value of the syndrom causes the error circuit to enable the correction circuit 38 to transfer the data without change to the memory bus. If the syndrom value indicates an error, signals are generated by the error circuit 37 to either correct the value of the data before transfer by the correction circuit 38 to the memory bus or to indicate that the data is incorrect. In one embodiment, the error circuit 37 generates a pattern of

bits which change a single bit of the sixty-four bits of data read from the memory 13.

Utilizing the circuit 30 of Figure 3, the process of writing to main memory 13 may be accelerated. Figure 4 is a diagram illustrating a number of write operations to a plurality of storage areas of a write buffer 33. The individual write operations affecting each storage area are described, and the results of those write operations are shown in Figure 4. Some of those operations are reviewed here

If valid data has been transferred to the write buffer 33 and is waiting to be written to memory after the controller finishes an earlier operation and additional valid data from a later write operation is received at the input to the write buffer 33 addressed to the same quad word address as the valid data already in the write buffer, then this additional valid data is transferred to the storage area already allotted to valid data to be written to the same address. This additional valid data overwrites any data (valid or not) at the bit positions of the incoming valid data in the storage area (see Figure 4, storage area holding data to address A). Thus, if all of the data in the addressed storage area is valid and new valid data is written to some or all of the bytes of the quad word addressed by the data in the storage area, the data residing in the storage area is updated by the new data; and all of the data in the storage area remains valid. Thus it will be seen, where two writes have occurred to the same quad word address space in memory and the first of these writes has not yet been written to memory when the second arrives, the result of both writes is combined before the write to memory so that only a single write of the latest valid data and a single computation of a check value is necessary.

The same result occurs if, after two writes have been combined, the controller is still not ready to write the data to memory 13. If any additional valid data addressed to that address is furnished to the write buffer 33, this data is written to the storage area containing the data already addressed to the same memory address. If this additional data

is addressed to byte positions presently storing valid data in the storage area, then the data in the storage area is stale and is by this technique continually updated in the write buffer 33 (see Figure 4, storage area holding data to addresses E and F). If the additional data to the write buffer 33 is addressed to byte positions presently storing data which is marked invalid, then the new data increases the amount of valid data in the addressed storage area and reduces the number of writes to memory (see Figure 4, storage area holding data to address C). This data merger reduces the time required to write by at least one write operation for each write merger in a system which uses error correcting code and, if all of the data in the storage area is valid when the controller 21 is ready to write to memory, will reduce what would have been a series of read/modify/write operations to a single simple write operation. This drastically reduces the time required to write data to memory. Not only does reducing all read/modify/write operations in a system using error correcting code to simple writes eliminate those repetitive steps, it also eliminates the need to test old check values and generate new check values during each read/modify/write operation of the individual words.

If, as in the example described above, four writes of words to sequential word positions in the same quad word occur in a burst, then all of these words may be merged in the quad word space of a storage area of the write buffer 33 before being written to memory presuming that the write controller is occupied with other operations for a long enough period (see Figure 4, storage area holding data to address D). Thus a burst of four write of individual words to sequential addresses of a quad word is reduced to a single write and the attendant generation of a single check value.

If valid data of any length exists in the write buffer 33 when the controller is ready to write to memory 13, then the operation begins without waiting. If the controller determines that all bytes of the quad word space of the storage area include valid data, the operation is a simple write of the data to memory along with a generation and write of a check

value to memory. When the write occurs, the data in the storage area of the write buffer from which the data has been written is invalidated. If less than all of the quad word space of the storage area includes valid data in a system using error correction code, then the controller 21 transfers the data with its status bits to the register 31 and initiates a read of the data stored at the addressed quad word space in memory. The valid bytes of the data which is read from memory are stored to those byte positions of the register 31 holding data marked invalid and merge with the valid bytes furnished by the multiplexor 32 to the register 31. Following these read and modify operations, the quad word in the register 31 which now contains all valid bytes is written to the addressed quad word space in memory; and the check value is generated from the data written and stored in memory.

When data is written in bursts, it is likely that the data of the individual writes will be addressed to adjacent words or double words of a single quad word space (or other sized memory address) of main memory. This will allow a significant amount of merging to occur. In such a case, the time saved in implementing writes is very substantial.

In order to implement a write buffer which accomplishes merger of writes to the same address in this manner, one embodiment utilizes an individual address comparator for each storage area in which a data increment may be stored. Valid data present at the input to the write buffer 33 has its address tested in parallel by the comparators at each storage area. If the write buffer 33 is empty, a new storage area is allocated; and the valid data is stored at that storage area with its address and status. Whenever a storage area is allocated, all of the status bits are set to match the valid bits of the data being written. If, however, the address compares with the address of valid data at any storage area, the new valid data is written to those bytes of that storage area. When new valid data is merged with valid data previously written to the storage area, only the status bits related to that new valid data are updated to indicate validity. If the address does not compare with the address of valid data in any storage area of the write buffer, the valid

data in the new write is stored in a newly allocated storage area in the write buffer; and all of the status bits are set to match the valid bits of the data being written. With each new write of valid data to the write buffer 33, the address is compared to the addresses of valid data entries. If the addresses are the same, the new valid data is merged with that entry; while if the address differs from the addresses of all data entries, a new storage area is allocated for the data. Once an entry has been written to memory, a global status bit for the storage area is marked invalid so that the space is free for reallocation. The individual byte status bits need not be invalidated since these will be updated to match new data when the storage area is again allocated.

In a write buffer with a plurality of storage areas, the valid data in the storage area which was first allocated is written first from the write buffer to memory (i.e., first in, first out). In order to accomplish this, in one embodiment the write buffer is a ring buffer which uses a pointer to the entry including the earliest data written to the register. Another way to implement this first-in first-out process would be to shift the data through from stage-to-stage of the buffer as each increment of data is written to memory.

In another embodiment of the invention, the hardware overhead required to allow data to be merged into any valid data in the write buffer 33 is reduced by allowing merger with only valid data in the last storage area to be allocated to a data entry. This eliminates the need for more than one comparator with the write buffer but requires that the valid data in the last storage area allocated be determinable. In the embodiment discussed using a pointer to the storage area first allocated, a pointer may also be furnished by which the most recently allocated storage area may be determined.

Figure 5 illustrates a comparison circuit 50 which may be used in such an embodiment to provide an indication that data in a new write is directed to the same address as data already stored in one of the storage areas. The circuit 50 illustrated includes at least that portion of

each of the storage areas 51-54 (only four storage areas are illustrated in the particular embodiment) used to stored the address of an area which has been allocated to data directed to a particular address. Each of these storage areas 51-54 is connected to receive the address to which any new write is being sent should the data of that write be ultimately stored in the storage area. Each of these storage areas 51-54 is also connected to furnish any address it presently stores as an input to a multiplexor 56. Included as a part of this address is a global status bit providing the validity status of the data in the storage area. The multiplexor is enabled to transfer the last address to which data has been written to a comparator circuit 55. The comparator 55 also receives the address of the incoming write (along with a status bit indicating that the data is valid) and compares the incoming address to the address stored in the last storage area to which data has been written in the write buffer. If the addresses are the same (including the status bit indicating that valid data exists in the storage area), there is a hit; and the data is addressed to the same storage area. If the addresses are not the same, there is no hit; and the data is sent to the next storage area in the write buffer.

In the embodiment of Figure 5, a register 57 is furnished a pointer to a next storage area available for allocation to data written to a new address so that the controller may know in which storage area to store data for which there is no address comparison. In Figure 5, for example, if the storage areas 53 and 54 include valid write data, the next storage area which could be allocated would be storage area 52 and the dotted line from the register 57 indicates the pointer to this storage area. If this pointer is stored in the register 57, a simple incrementing on the pointer value provides a value indicating the last storage area to which valid data was written so that the value in the register 57 may be used to enable the multiplexor 56 to transfer the address of the data in the storage area 53 to the comparator 55.

In any embodiment of the invention, the same general rules obtain. Writes to memory from storage areas of the write buffer occur on a first-in

(allocated) first-out basis. Merger with data in an allocated storage area or allocation of a new storage area for new data written to the buffer is determined by whether valid bytes exist at an entry having the same address. No write to memory is stalled to allow a merger to occur before the write takes place.

However, in some embodiments of the invention, other rules of storage in the write buffer and writing to memory are enforced. Thus, for example, if new valid data is written to a storage area from which data is to be written to memory in the same clock in which it would be merged, a number of boundary and speed path conditions are raised. For example, a decision whether to conduct a simple write or a read/modify/write must be made by the controller 21 before the data is written to memory; if a decision has already been made and then new data is merged in the storage area before the write, the result of the decision may be changed. In order to eliminate these boundary and speed path conditions, valid data is not written to a storage area from which data is to be written to memory in the same clock in which it would be merged. A consequence of enforcing this rule is that in an embodiment including a single comparator, data may not be merged with valid data in a storage area containing the only valid data in the write buffer; whether this condition exists may be determined by testing the condition of the pointers to the data next to be written to memory and to the storage area next to be written.

In one embodiment of the invention, a global valid bit and valid bits for each byte are provided for each data entry in a storage area of the write buffer. Then, when a write to memory occurs, only the global valid bit is invalidated. When the storage area is reallocated to data on a later write, the individual status bits are set in accordance with each byte of the data being stored during that entry.

Although the present invention has been described in terms of a preferred embodiment, it will be appreciated that various modifications and alterations might be made by those skilled in the art without

departing from the spirit and scope of the invention. The invention should therefore be measured in terms of the claims which follow.

What Is Claimed Is:

Claim 1. A storage device for buffering data being transferred on a memory bus from a source of data to memory comprising:

a plurality of storage areas each capable of storing a memory bus width of data and a memory address for data stored in the area;

input circuitry for writing new valid data furnished by the source of data to a storage area in place of data previously written to a memory address to which the new valid data is addressed;

allocation circuitry for storing in an unused storage area new data being written to a memory address different than a memory address stored in any storage area;

a memory controller for writing data to memory from a storage area in bus width increments, the memory controller reading valid data at any memory address to which less than a bus width of valid data is being written from a storage area and modifying the data read with the valid data from the storage area before writing a bus width increment of data to memory, and

error correction code generating circuitry for generating and storing in memory an error correction value unique to each bus width increment of data written to memory.

Claim 2. A storage device as claimed in Claim 1 in which the input circuitry for writing new valid data furnished by the source of data to a storage area in place of data previously written to a memory address to which the new valid data is addressed comprises comparator circuitry for comparing a memory address of new data with memory addresses of data stored in storage areas.

Claim 3. A storage device as claimed in Claim 2 in which the comparator circuitry compares a memory address of new data only with a memory address of data stored in a last allocated storage area.

Claim 4. A storage device as claimed in Claim 1 in which the memory controller writes valid data to memory from a storage area allocated to data least recently.

Claim 5. A storage device as claimed in Claim 4 in which the memory controller maintains a pointer to a storage area allocated to data least recently.

Claim 6. A storage device as claimed in Claim 1 in which the input circuitry for writing new valid data furnished by the source of data to a storage area in place of data previously written to a memory address to which the new valid data is addressed continues writing new valid data to a storage area until the memory controller begins to write data to memory from the storage area.

Claim 7. A storage device as claimed in Claim 6 in which a write of data to memory from a storage area by the memory controller is never delayed to allow the input circuitry to write new valid data furnished by the source of data to the storage area.

Claim 8. A storage device for buffering data being transferred on a memory bus from a source of data to memory comprising:

a plurality of storage areas each capable of storing data equal to a memory bus width and a memory address for the data,

means for writing valid bits of new data to a storage area containing valid data addressed to a memory address of the new data in place of any bits of data already in the storage area,

means for writing new data being written to a memory address different than any memory address stored in any storage area to an unused storage area; and

means for writing to memory all valid data in a storage area when memory is able to store data and generating an error correction value

for each bus width of data at a memory address to which the data is being written.

Claim 9. A storage device as claimed in Claim 8 in which the means for writing valid bits of new data to a storage area containing valid data addressed to a memory address of the new data in place of any bits of data already in the storage area comprises comparator circuitry for comparing a memory address of new data with memory addresses of data stored in storage areas.

Claim 10. A storage device as claimed in Claim 9 in which the comparator circuitry compares a memory address of new data only with a memory address of data stored in a last allocated storage area.

Claim 11. A storage device as claimed in Claim 8 in which the means for writing to memory all valid data in a storage area writes valid data to memory from a storage area allocated to data least recently.

Claim 12. A storage device as claimed in Claim 11 in which the means for writing to memory all valid data in a storage area includes means for maintaining a pointer to a storage area allocated to data least recently.

Claim 13. A storage device as claimed in Claim 8 in which the means for writing valid bits of new data to a storage area containing valid data addressed to a memory address of the new data in place of any bits of data already in the storage area continues writing new data to a storage area until the means for writing to memory all valid data in a storage area begins to write data to memory from the storage area.

Claim 14. A storage device as claimed in Claim 13 in which a write of data to memory from a storage area by the means for writing to memory all valid data in a storage area is never delayed to allow the means for writing valid bits of new data to a storage area containing valid data addressed to a memory address of the new data to write new valid data to the storage area.

Claim 15. A computer system comprising:

a central processing unit;

main memory;

a memory bus for transferring data from the central processing unit to main memory;

a write buffer on the memory bus for buffering data being transferred on the memory bus from a the central processing unit to main memory including:

a plurality of storage areas each capable of storing a memory bus width of data and a memory address for data stored in the area;

input circuitry for writing new valid data furnished by the central processing unit to a storage area in place of data previously written to a memory address to which the new valid data is addressed;

allocation circuitry for storing in an unused storage area new data being written to a memory address different than a memory address stored in any storage area;

a memory controller for writing data to main memory from a storage area in bus width increments, the memory controller reading valid data at any memory address to which less than a bus width of valid data is being written from a storage area and modifying the data read with the valid data from the storage area before writing a bus width increment of data to main memory, and

error correction code generating circuitry for generating and storing in main memory an error correction value

unique to each bus width increment of data written to main memory.

Claim 16. A computer system as claimed in Claim 15 in which the input circuitry for writing new valid data furnished by the central processing unit to a storage area in place of data previously written to a memory address to which the new valid data is addressed comprises comparator circuitry for comparing a memory address of new data with memory addresses of data stored in storage areas.

Claim 17. A computer system as claimed in Claim 15 in which the memory controller writes valid data to main memory from a storage area allocated to data least recently.

Claim 18. A computer system as claimed in Claim 15 in which the input circuitry for writing new valid data furnished by the source of data to a storage area in place of data previously written to a memory address to which the new valid data is addressed continues writing new valid data to a storage area until the memory controller begins to write data to main memory from the storage area.

Claim 19. A method for buffering writes of data from a source of data to memory in a system in which each write of data to memory is of a full memory bus width of data and error correction code is computed for each write using a full memory bus width of data and stored in memory, the method comprising the steps of:

storing new data from the source of data first written to a memory address in an unused one of a plurality of storage areas in a write buffer with the memory address,

storing new data from the source of data to a storage area in place of data already written to the storage area for a memory address of the new data, and

writing to memory all valid data in a storage area when memory is able to store data and generating an error correction value for each bus width of data at a memory address to which the data is being written.

Claim 20 A method for buffering writes of data from a source of data to memory as claimed in Claim 19 in which:

the step of storing new data from the source of data to a storage area in place of data already written to the storage area for a memory address of the new data continues until the step of writing to memory all valid data in a storage area begins, and

the step of writing to memory all valid data in a storage area comprises writing valid data to memory from a storage area allocated to data least recently.

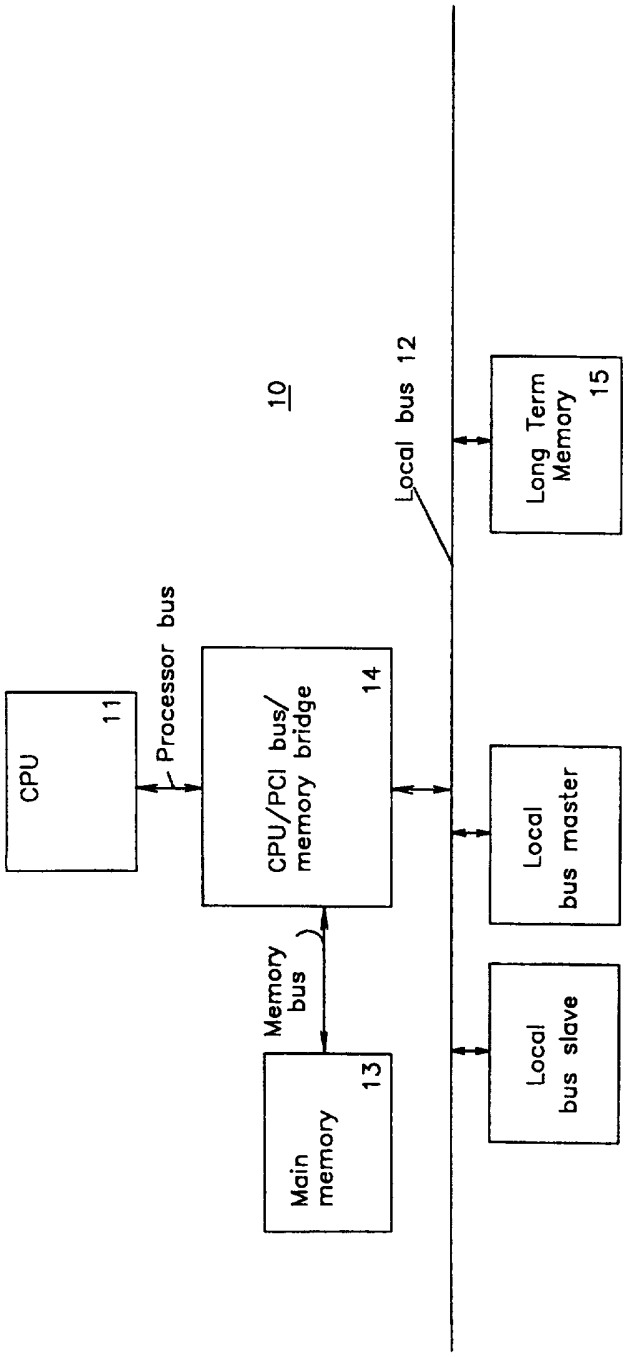

10
Local bus 12
CPU
11
Processor bus
CPU/PCI bus/
memory bridge
14
Memory
bus
Main
memory
13
Long Term
Memory
15
Local
bus master
Local
bus slave

Figure 1

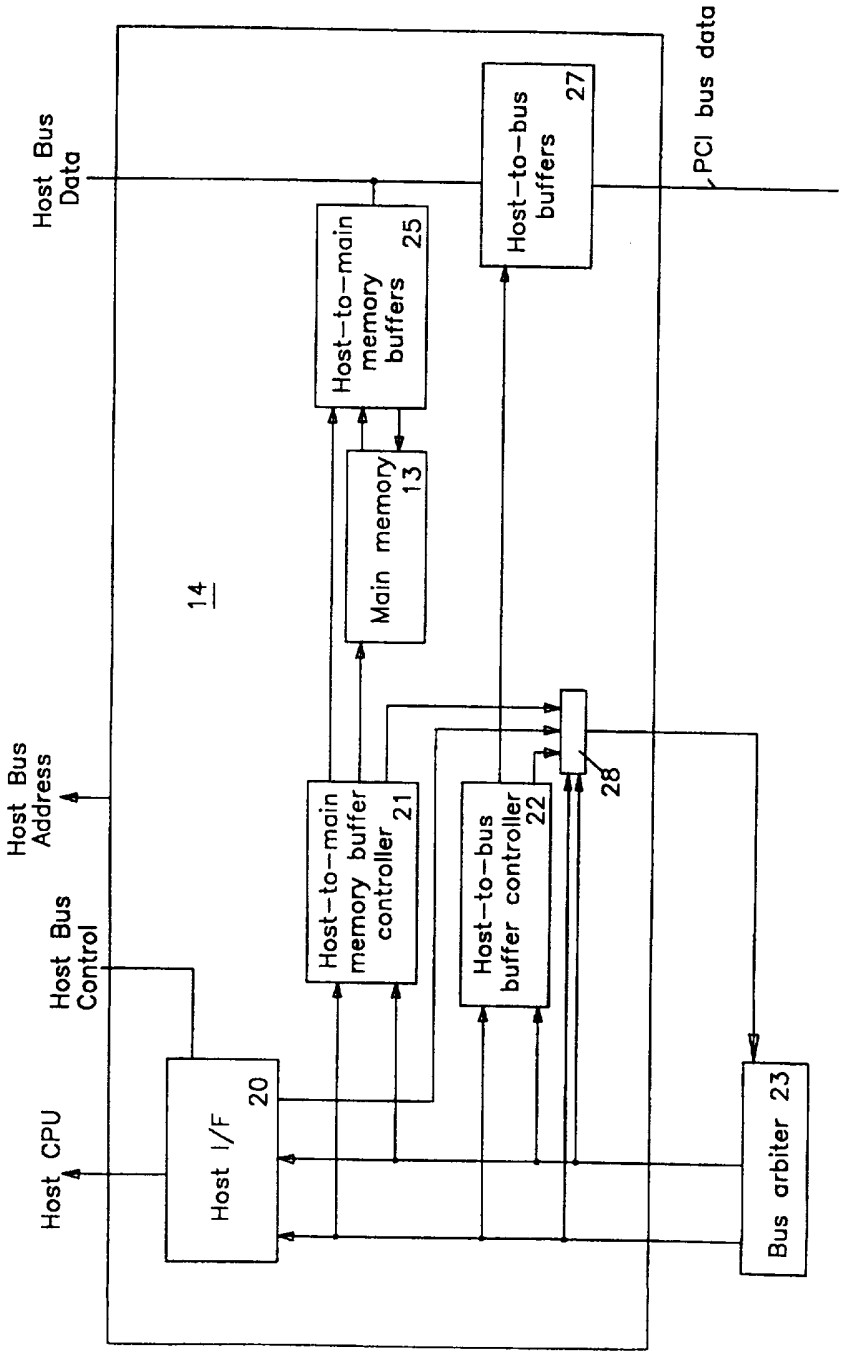

Host Bus Data
PCI bus data
Host-to-bus buffers 27
Host-to-main memory buffers 25
Main memory 13
14
Host Bus Address
Host Bus Control
Host-to-main memory buffer controller 21
Host-to-bus buffer controller 22
28
Host CPU
Host I/F 20
Bus arbiter 23

Figure 2

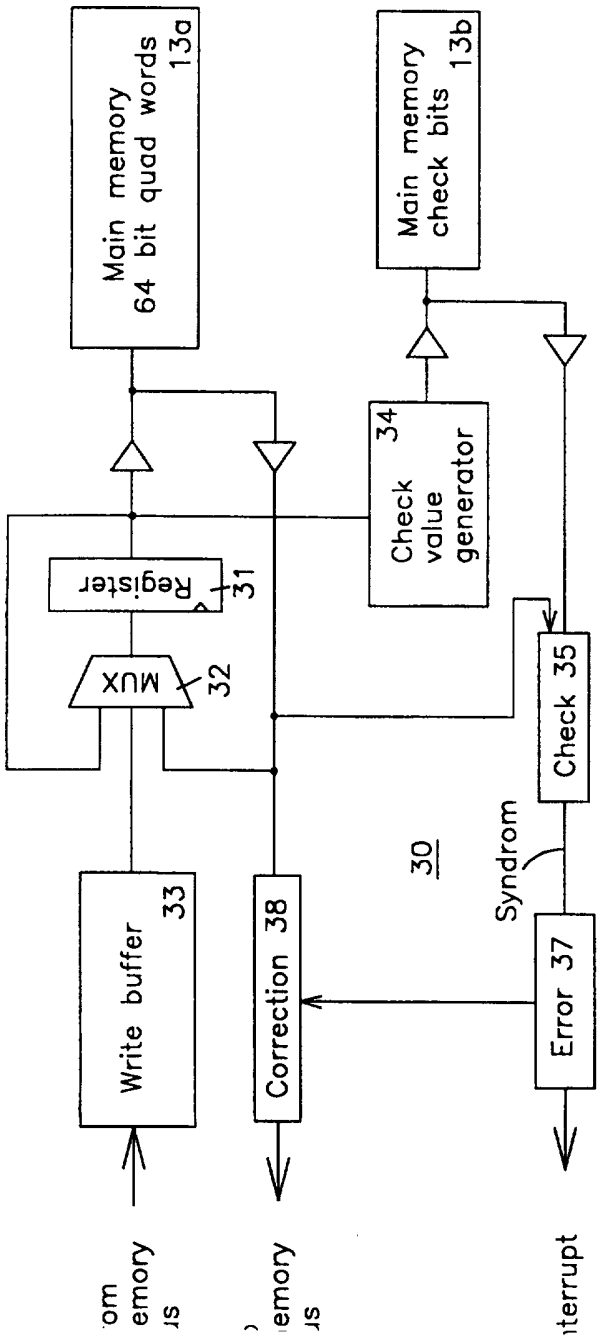

Main memory
64 bit quad words
13a
Main memory
check bits
13b
Check
value
generator
34
Register
31
MUX
32
Check 35
Syndrom
30
Write buffer
33
Correction 38
Error 37
om
emory
ıs
emory
ıs
ıterrupt

Figure 3

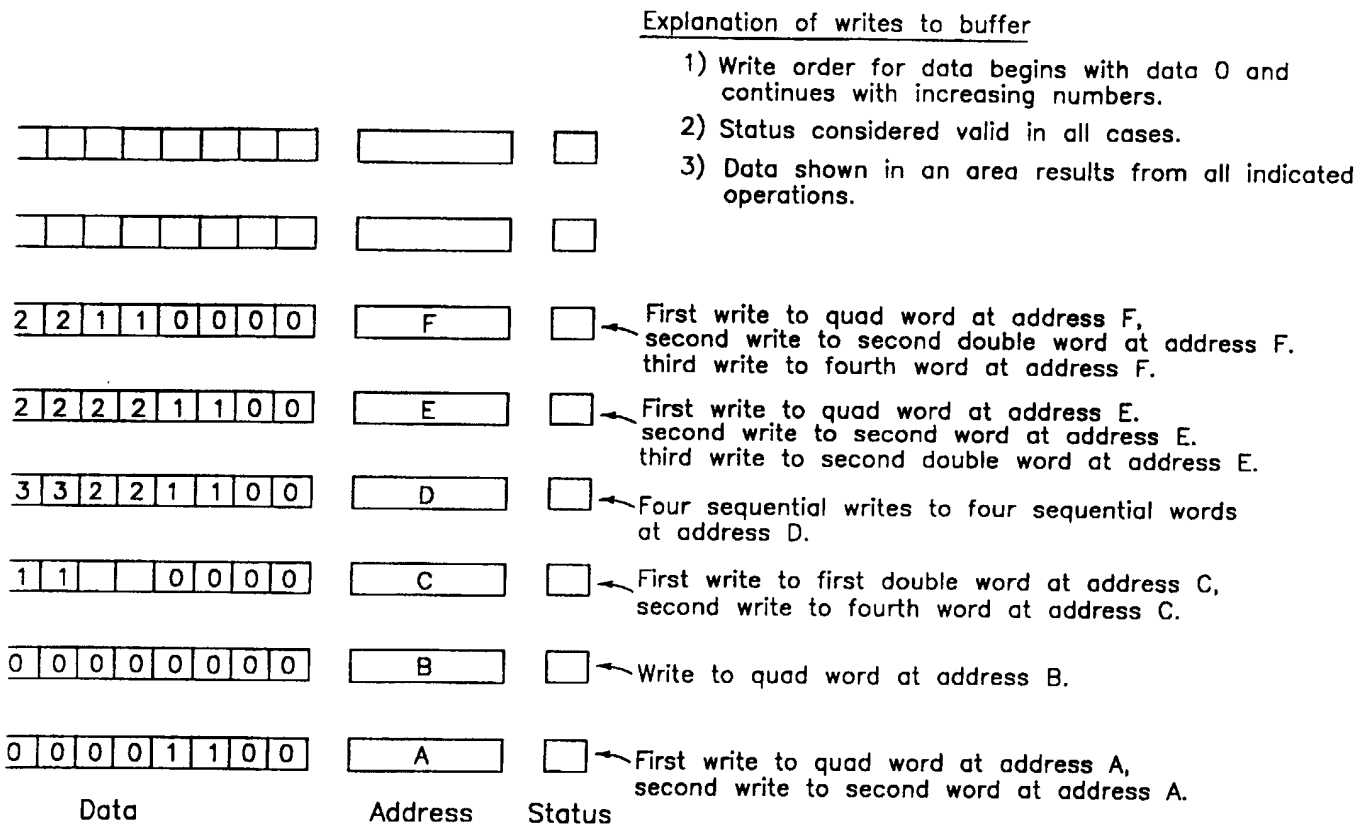

Explanation of writes to buffer
1) Write order for data begins with data 0 and continues with increasing numbers.
2) Status considered valid in all cases.
3) Data shown in an area results from all indicated operations.
2 2 1 1 0 0 0 0
F
First write to quad word at address F, second write to second double word at address F. third write to fourth word at address F.
2 2 2 2 1 1 0 0
E
First write to quad word at address E. second write to second word at address E. third write to second double word at address E.
3 3 2 2 1 1 0 0
D
Four sequential writes to four sequential words at address D.
1 1 0 0 0 0
C
First write to first double word at address C, second write to fourth word at address C.
0 0 0 0 0 0 0 0
B
Write to quad word at address B.
0 0 0 0 1 1 0 0
A
First write to quad word at address A, second write to second word at address A.
Data
Address
Status

Figure 4

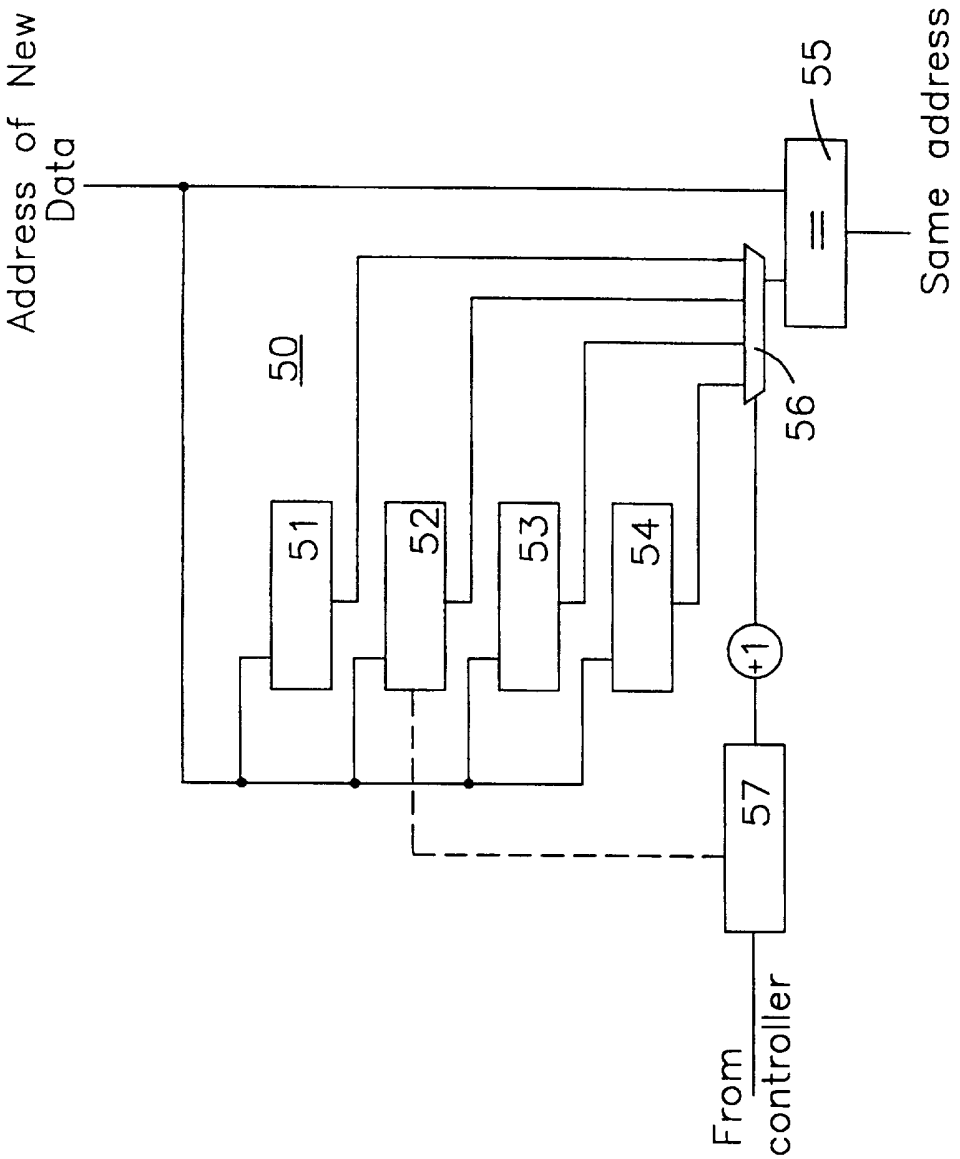

Address of New Data
50
51
52
53
54
55
=
Same address
56
+1
57
From controller

Figure 5

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US96/11840

A. CLASSIFICATION OF SUBJECT MATTER
IPC(6) :G01R 31/28; G06F 11/00
US CL :395/182.03
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)

U.S. : 371/ 40.1, 40.2, 48, 51.1; 395/180, 181, 182.03, 182.04, 427

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 practicable, search terms used)

APS, CAS ONLINE

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US, A, 5,285,451 (HENSON ET AL.) 08 February 1994, see the abstract, col. 7, line 64 to col. 9, line 35, col. 13, line 40 and et seq.	1-20
Y	US, A, 5,276,849 (PATEL) 04 January 1994, see the abstract, figure 2, col. 2, lines 38-67, col. 3, lined 48-60	1-20
Y,E	US, A, 5,542,065 (BURKES ET AL.) 30 July 1996, see the abstract and the entire document	1-20
Y	US,A, 5,214,777 (CURRY, JR.) 25 May 1993, see the abstract, also col.. 2, lines 36-61, col. 4, lines 24-39	1-20

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

*	Special categories of cited documents:	"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
"A"	document defining the general state of the art which is not considered to be part of particular relevance	"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
"E"	earlier document published on or after the international filing date	"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
"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)	"&"	document member of the same patent family
"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		

Date of the actual completion of the international search
29 AUGUST 1996

Date of mailing of the international search report
18 SEP 1996

Name and mailing address of the ISA/US
Commissioner of Patents and Trademarks
Box PCT
Washington, D.C. 20231
Facsimile No. (703) 305-3230

Authorized officer
ALBERT DECADY
Telephone No. (703) 308-3800

Form PCT/ISA/210 (second sheet)(July 1992)*

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US96/11840

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US,A, 5,235,693 (CHINNASWAMY ET AL.) 10 August 1993, see the abstract, figures 2-4, col. 1, lines 55-68, col. 3, lines 23-51	1-20