634354

COMMONWEALTH of AUSTRALIA
Patents Act 1952

APPLICATION FOR A STANDARD PATENT

I/We

Sun Microsystems, Inc.

of

2550 Garcia Avenue, Mountain View, California, 94043, United States of America

hereby apply for the grant of a Standard Patent for an invention entitled:

Method and apparatus for predictive caching

which is described in the accompanying complete specification.

Details of basic application(s):-

Number	Convention Country	Date
576672	United States of America	31 August 1990

The address for service is care of DAVIES & COLLISON, Patent Attorneys, of 1 Little Collins Street, Melbourne, in the State of Victoria, Commonwealth of Australia.

DATED this FIRST day of MARCH 1991

To: THE COMMISSIONER OF PATENTS

.....
a member of the firm of
DAVIES & COLLISON for
and on behalf of the
applicant(s)

Davies & Collison, Melbourne

COMMONWEALTH OF AUSTRALIA

PATENTS ACT 1952

DECLARATION IN SUPPORT OF CONVENTION OR NON-CONVENTION APPLICATION FOR A PATENT

In support of the Application made for a patent for an invention entitled: "Method and apparatus for predictive caching"

Insert title of invention

I/We Michael H. Morris of SUN MICROSYSTEMS, INC. of 2550 Garcia Avenue Mountain View, California 94043 United States of America

Insert full name(s) and address(es) of declarant(s) being the applicant(s) or person(s) authorized to sign on behalf of an applicant company.

do solemnly and sincerely declare as follows :-

Cross out whichever of paragraphs 1(a) or 1(b) does not apply

1(a) relates to application made by individual(s)
1(b) relates to application made by company; insert name of applicant company.

1. (a) I/We am/are the applicant(s) for the patent

or (b) I am authorized by

Sun Microsystems, Inc.

the applicant..... for the patent to make this declaration on its/their behalf.

Cross out whichever of paragraphs 2(a) or 2(b) does not apply

2(a) relates to application made by inventor(s)
2(b) relates to application made by company(s) or person(s) who are not inventor(s), insert full name(s) and address(es) of inventors.

2. (a) I am/We are the actual inventor(s) of the invention

or (b) 1. Curtis Priem of 4052 Kettering Terrace, Fremont, California 94536
2. Chris Malachowsky of 3775 Thrush Way, Santa Clara, California 95051
3. Robert Rocchetti of 10870 N. Stelling Road, Apt. 30C, Cupertino, California 95014
and 4. David Rosenthal of 427 Alma Street, Apt. 407, Palo Alto, California 94301
All of the United States of America

is/are the actual inventor..s..... of the invention and the facts upon which the applicant.....

is/are entitled to make the application are as follows :-

The actual inventors have assigned the invention to the said applicant.

State manner in which applicant(s) derive title from inventor(s)

Cross out paragraphs 3 and 4 for non-convention applications. For convention applications, insert basic country(s) followed by date(s) and basic applicant(s).

3. The basic application..... as defined by Section 141 of the Act was/were made in United States of America on the 31 August 1990 by Curtis Priem, Chris Malachowsky, Robert Rocchetti and/& David Rosenthal ..... on the .....

by .....

in ..... on the .....

by .....

4 The basic application..... referred to in paragraph 3 of this Declaration was/were the first application..... made in a Convention country in respect of the invention the subject of the application.

Insert place and date of signature.

Declared at Palo Alto, Calif. this 4th day of February 1991

Signature of declarant(s) (no attestation required)

Michael H. Morris
Vice President

Note: Initial all alterations

DAVIES & COLLISON, MELBOURNE and CANBERRA.

AU9172030

(12) PATENT ABRIDGMENT (11) Document No. AU-B-72030/91
(19) AUSTRALIAN PATENT OFFICE (10) Acceptance No. 634354

(54) Title
METHOD AND APPARATUS FOR PREDICTIVE CACHING

International Patent Classification(s)
(51)[5] **G06F 012/00**

(21) Application No. : **72030/91** (22) Application Date : **01.03.91**

(30) Priority Data

(31) Number	(32) Date	(33) Country
576672	**31.08.90**	**US UNITED STATES OF AMERICA**

(43) Publication Date : **05.03.92**

(44) Publication Date of Accepted Application : **18.02.93**

(71) Applicant(s)
SUN MICROSYSTEMS, INC.

(72) Inventor(s)
CURTIS PRIEM; CHRIS MALACHOWSKY; ROBERT ROCCHETTI; DAVID ROSENTHAL

(74) Attorney or Agent
DAVIES COLLISON CAVE , 1 Little Collins Street, MELBOURNE VIC 3000

(56) Prior Art Documents
GB 1515376
US 3898624
EP 109309

(57) Claim

1. A predictive caching system for use in computer system having at least one portion of memory in which information is stored for retrieval, and a processor for controlling the access of the memory comprising means for discerning a pattern of access of the memory, means operating in response to the pattern determined by the means for discerning a pattern of access of the memory for determining a next address which will probably be accessed in such memory if the pattern discerned continues, and means for storing the information at the next address determined prior to the next access of the memory whereby the information at the next address is available without the need to access the memory.

2. A predictive caching system for use in computer system as claimed in Claim 1 in which the means for discerning a pattern of access of the memory comprises means for storing information from the last address accessed and the addresses immediately surrounding the last address accessed in the one portion of memory.

634354

COMMONWEALTH OF AUSTRALIA

PATENTS ACT 1952

COMPLETE SPECIFICATION

NAME & ADDRESS
OF APPLICANT:

Sun Microsystems, Inc.
2550 Garcia Avenue
Mountain View California 94043
United States of America

NAME(S) OF INVENTOR(S):

Curtis PRIEM
Chris MALACHOWSKY
Robert ROCCHETTI
David ROSENTHAL

ADDRESS FOR SERVICE:

DAVIES & COLLISON
Patent Attorneys
1 Little Collins Street, Melbourne, 3000.

COMPLETE SPECIFICATION FOR THE INVENTION ENTITLED:

Method and apparatus for predictive caching

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

BACKGROUND OF THE INVENTION

1. Field of the Invention:

This invention relates to caching arrangements used in computer systems and, more particularly, to methods and apparatus for caching data expected to be used by such systems.

2. History of the Prior Art:

In computer systems, the access of main memory to retrieve information often takes a substantial portion of the operational time. This occurs for various reasons. First, main memory is random access memory which is often sufficiently large that its cost is kept within bounds by using relatively slow memory devices. Second, in systems using virtual memory arrangements to increase the addressable space without increasing the physical size of the main memory, it is necessary to address information in both main memory and secondary memory such as magnetic disk storage by means of memory management arrangements which utilize look-up tables to translate virtual addresses to physical addresses. This indirect addressing requires a substantial amount of time.

For this reason, the use of cache memories to increase system speed has become prevalent in more advanced systems. A cache memory makes use of a relatively small amount of fast (and therefore expensive) random access memory in which recently used instructions, data, or both are stored as they are used by a processor. Such information is then available in the cache memory so that it may be more rapidly accessed by the associated processor

when next required. The basic theory of caching is that, in general, information which has been recently used is more likely to be used sooner than is other information. The cache memory is often both physically faster than the random access memory used for main memory and is arranged so that it may be addressed more rapidly than may main memory by obviating the need for the look-up operation to determine physical addresses. Such caching arrangements have operated to greatly increase the speed of operation of computer systems for certain types of information.

Caching works well in situations where the same information is repeatedly utilized in carrying out an operation. An example of such a use is found in executing an instruction including a looping operation. Caching does not usually work well in situations in which entirely new information is used in each step of the operation. For example, in scrolling down the screen of a computer output display which shows approximately one thousand pixels of eight bit bit-mapped color information in each of the horizontal and vertical directions, approximately one megabyte of data is traversed. A typical cache memory may hold sixty-four kilobytes of information. A scrolling operation essentially reads information in a first scan line and writes that information to the next line below or above. When the cache is used to accomplish scrolling, it can hold only a few lines from such a frame buffer. Thus, except for the first few lines, none of the information which has been placed in the cache memory is reused so no increase in speed is accomplished. Moreover, since for the greatest part of the scrolling operation the processor is continually filling the cache memory with new information, the operation takes much longer using a cache memory than it would without a cache memory. In such a cache fill

operation, the processor must first look to the cache memory for the information, determine that the information is missing in the cache memory, go to the look-up tables for the physical address of the information, access the information to be used in main memory, store that information in the cache memory replacing other information, and finally use the information. Such an operation must repeat continuously during a scrolling operation so that the fact that the operation takes longer than a simple access of main memory should not be surprising.

One method of increasing the speed of operation of a computer is to design the processor to handle instructions which are pipelined so that one instruction is executed during each clock cycle of operation. While the early pipelined computer systems using caching demonstrated a loss of approximately four instructions for each cache memory miss, the faster systems being designed at present will have a loss of as much as sixty-four instruction times for each cache miss. This is a very large delay in a fast computer system. For this reason, many systems now include methods for disabling the caching operation in situations in which it may delay the operation of the system. Although such methods eliminate the delays caused by the caching of certain information, they do not speed the handling of that information in any other way.

SUMMARY OF THE INVENTION

According to the present invention there is provided a predictive caching system for use in computer system having at least one portion of memory in which information is stored for retrieval, and a processor for controlling the access of the memory comprising means for discerning a pattern of access of the memory, means operating in response to the pattern determined by the means for discerning a pattern of access of the memory for determining a next address which will probably be accessed in such memory if the pattern discerned continues, and means for storing the information at the next address determined prior to the next access of the memory whereby the information at the next address is available without the need to access the memory.

The invention is further described by way of example only with reference to the accompanying drawings in which like elements are referred to by like designations throughout the several views. In the drawings:

AUSTRALIAN PATENT OFFICE LS

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is an illustration of a frame buffer showing various elements which may be stored therein for display.

Figure 2 is an enlargement of a portion of the frame buffer illustrated in Figure 1 showing the various elements which may be stored therein for display.

Figure 3 is a block diagram illustrating a circuit for accomplishing caching in accordance with the invention.

Figure 4 is an illustration of a portion of a frame buffer useful in understanding the operation of this invention.

Figure 5 is a block diagram illustrating a second circuit for accomplishing caching in accordance with the invention.

NOTATION AND NOMENCLATURE

Some portions of the detailed descriptions which follow are presented in terms of algorithms and symbolic representations of operations on data bits within a computer memory. These algorithmic 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.

An algorithm is here, and generally, conceived to be a self-consistent sequence of steps leading to a desired result. The steps 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 apparatus and to method steps for operating a computer in processing electrical or other (e.g. mechanical, chemical) physical signals to generate other desired physical signals.

DETAILED DESCRIPTION OF THE INVENTION

Referring now to Figure 1, there is shown a diagram of a frame buffer 10 used in a typical work station. The frame buffer 10 may be considered for the purpose of this specification to be capable of holding data for display on a bit-mapped computer output display having approximately one thousand pixels along each scan line in a horizontal direction and approximately one thousand scan lines in a vertical direction. In general, the exemplary frame buffer 10 may be considered to hold eight bits of information for each pixel so that it may hold in total approximately one megabyte of data for screen display.

In general, for the purpose of this discussion, each pixel of information may be considered to be stored in a position in the frame buffer 10 which is essentially the same position as that in which it will appear on the output display. Although this is not necessary true, it is at least a correct logical statement and probably makes understanding somewhat easier. Therefore, those elements which are described by the information stored in the frame buffer 10 are illustrated as stored in positions in the frame buffer 10 which are the positions in which they would be displayed on the computer output display. For example, the vectors or line segments 12, 13, and 14 illustrated in Figure 1 are stored in pixel positions (whether logical or physical) of the frame buffer 10 which are the positions in which they appear on the display screen.

Figure 2 illustrates a small (ten by ten pixels) enlarged portion 15 of the frame buffer 10 illustrated in Figure 1 containing portions of each of the lines 12, 13, and 14 shown in Figure 1. That portion of the line 13 shown in Figure 2

is described by ten pixels stored serially adjacent one another on a single horizontal row of the frame buffer. This information will appear on a single scan line of the output display. That portion of the line 14 shown in Figure 2 is described by ten pixels stored adjacent above and below one another in a single column of a frame buffer which pixels will be described on ten adjacent scan lines of the output display. That portion of the line 12 shown in Figure 2 is described by pixels stored on a number of adjacent horizontal rows of the frame buffer to be shown on a number of adjacent scan lines of the output display, the pixels on each scan line lying serially adjacent one another.

The three line segments 12, 13, and 14 are typical of the various elements which are placed in a frame buffer for display on a bit-mapped output display of a computer system. Such elements may be used to construct larger figures of almost any shape. Reviewing the three different line segments 12, 13, and 14, those skilled in the art will recognize that when the line 13 is stored in the frame buffer, the pixels lie in the same row with linear addresses immediately adjacent one another. It is usual for a system in filling a general system cache memory to write not only the specific information accessed but also information adjacent to that accessed to the cache memory because such information is likely to be accessed quite soon. Thus, once any position on the line 13 has been accessed, the information defining the pixels adjacent to the accessed position on that line will usually be held within a typical cache associated with a processor so that those pixels may also be accessed without occasioning other than a start up cache miss.

On the other hand, the line 14 is described by pixels each of which has an address which has the same X value but is one row down from the last row accessed. Depending on the size of the cache and operation being performed, it is typical that only the pixels immediately adjacent each other on the same row will be held within the cache after a first access of a position on the line 14 when accomplishing an update of the display. Consequently, when the line 14 is accessed, a cache miss will probably occur with each pixel of the line 14 in reading the cache memory to write the line 14 to the output display. Finally, the pixels describing the line 12 have addresses immediately adjacent each other on individual rows but offset in the Y direction by one pixel from row to row. As with the line 14, only pixels of line 12 in the same row as an accessed pixel will usually reside within the typical cache. Thus, when the line segment 12 is written to the output display, a cache miss will occur between each scan line, three misses in total for the portion of the line segment 12 illustrated in Figure 2.

However, when either reading or writing any of the lines shown in Figure 2, the next pixel to be read or written will be a pixel having an address within one pixel of the last described pixel in either the X or the Y direction (or both) within the frame buffer. Superimposed on Figure 2 is a box 17 illustrating such a group of nine adjacent pixels stored in a frame buffer. If the pixel presently being accessed to describe a line segment on the display is considered to lie in the center position of the nine pixels in the box 17, then the next pixel to be accessed will lie in one of the other eight pixel positions within that box 17 so long as the same line segment is being described. This is true for both read and write accesses of the frame buffer where any connected line segment is involved.

On the other hand, the line 14 is described by pixels each of which s an address which has the same X value but is one row down from the last row accessed. Depending on the size of the cache and operation being performed, it is typical that only the pixels immediately adjacent each other on the same row will be held within the cache after a first access of a position on the line 14 when accomplishing an update of the display. Consequently, when the line 14 is accessed, a cache miss will probably occur with each pixel of the line 14 in reading the cache memory to write the line 14 to the output display. Finally, the pixels describing the line 12 have addresses immediately adjacent each other on individual rows but offset in the Y direction by one pixel from row to row. As with the line 14, only pixels of line 12 in the same row as an accessed pixel will usually reside within the typical cache. Thus, when the line segment 12 is written to the output display, a cache miss will occur between each scan line, three misses in total for the portion of the line segment 12 illustrated in Figure 2.

However, when either reading or writing any of the lines shown in Figure 2, the next pixel to be read or written will be a pixel having an address within one pixel of the last described pixel in either the X or the Y direction (or both) within the frame buffer. Superimposed on Figure 2 is a box 17 illustrating such a group of nine adjacent pixels stored in a frame buffer. If the pixel presently being accessed to describe a line segment on the display is considered to lie in the center position of the nine pixels in the box 17, then the next pixel to be accessed will lie in one of the other eight pixel positions within that box 17 so long as the same line segment is being described. This is true for both read and write accesses of the frame buffer where any connected line segment is involved.

Consequently, for each of the read and write operations in a frame buffer involving a line segment, the address of a next pixel may be accurately predicted so long as a line segment of any shape is being described. Only when a line segment begins or terminates is the position of the next pixel not accurately predictable. Since the next pixel address in describing a line may be accurately predicted, a small cache may be used to store the data at the addresses of the next possible pixels for describing any line segment. For the two operations described (read and write), such a cache need only hold the data for the position presently being accessed and the eight pixel positions surrounding the present position to assure that a cache miss will not occur on the next access.

Although the size of such a cache is trivial (nine memory storage locations), the increase in speed produced is not. For example, in reading the pixels for the line 14 from the frame buffer so that they may be written to the output display when using a normal cache associated with the central processing unit, it can be expected that a cache miss will occur for each pixel. If a line segment to be described is ten pixels long, then it can be expected that delays in a pipelined architecture of up to sixty-four instruction times may occur ten times in describing such a line. Using the small cache herein described, however, a cache miss will probably occur only on the first pixel of the line 14 thereby speeding the operation by a total of sixty-four instructions times multiplied by nine. This is a great deal of acceleration for a very small cache to accomplish. It has been determined through statistical studies that read accesses are approximately fifty percent of the frame buffer operations. Some operations may be accelerated substantially by the use of a cache memory as

herein described because the full access of each pixel position in the frame buffer may be replaced by the access of the more-rapidly operating cache memory. This result assumes that the cache can remain filled by transparently performing frame buffer accesses between host CPU frame buffer access requests.

To generalize, there are three types of frame buffer raster operations which may take place. These are operations which may be classified as zero, one, and two operand raster operations. A zero operand raster operation is a write only operation; an example is clearing an area of the screen. Since there is no need to know what was held in the particular pixels of the frame buffer, no cache is needed for the next operation. However, a small cache can speed up the operation on write accesses by essentially providing a level of write buffering. The cache can accept the write access, capture the relevant data, and release the host. The host assumes the write has taken place and continues with other operations. In the meantime the cache actually writes the data to the frame buffer. An example of a one operand operation is an invert command in which a pixel is changed from black to white. Such an operation requires a read and a write for each pixel, for a line segment, sufficient storage is provided by the small cache described above for such an operation. A two operand operation, on the other hand, may be exemplified by a copy operation such as a vertical scroll of the display. In such an operation, the information at one pixel is read in a first location and written into a second location. For any line segment, a cache in accordance with this invention must be large enough to store information regarding the nine addresses described for the location to be read plus another nine addresses storing the pixels to be written. That is, a

pixel is read and written to a new address a directed distance away, for example, one scan line directly below. Although accessing the first address to be read will cause a cache miss, the second address to be written is known from the directed distance given by the instruction. Moreover, the next address to be read will be within the eight pixels surrounding the first accessed read position while the next address to be written lies at the same directed distance from the second pixel read as did the first pixel written from the first pixel read. Thus, presuming these addresses are cached, sufficient information is available to determine each position; and no further cache misses will occur until the copying of the line segment ends.

Thus it is clear that the small caching arrangement proposed by this invention offers substantial increases in speed for the operation of frame buffers in all but the simplest zero operand operations.

The construction in hardware of such an arrangement requires only sufficient cache memory to record the data and addresses in at least the eighteen individual positions including the initial position from which a pixel may be read, the eight pixels surrounding the initial position, the first position to which a pixel is to be copied, and the eight pixels surrounding that first position. A circuit arrangement such as described is illustrated in block diagram form in Figure 3. The arrangement illustrates a central processor 22, a main memory 24, a frame buffer 26, and a small predictive cache memory 28. Each of the elements is connected to the system bus. The cache memory 28 is placed at the bus interface so that it can acknowledge the existence of addressed information without the necessity of accessing the frame buffer 28. The

arrangement of Figure 3 is adapted to be controlled by system software so that any access of the frame buffer 26 by the central processor 22 causes the information accessed and its address to be placed in the central position of the appropriate nine pixel positions of the cache 28. Then the information stored in the eight addresses of the frame buffer 26 surrounding the accessed position are placed with their addresses in the other eight positions of the cache memory 28. In general, this filling requires three or four accesses of the frame buffer. Although this filling of the cache 28 takes some time, the overall system latency usually provides sufficient time for the fill to be accomplished between frame buffer accesses. If the operation is such that the information is to be written to another position of the frame buffer 26, the address of that position is determined and placed in the center of the second group of nine memory spaces in the cache memory 28. From that point on, the information regarding that line segment will be available for access in the cache 28. On the next access of an adjoining pixel, the information and address of that adjoining pixel is placed in the central position of the first group of nine pixels and the other positions are filled from the addresses surrounding that adjoining pixel, many of which are already in the cache. In like manner, the information for the new position is moved to the center of the second nine pixels as each pixel is written. A cache memory 28 designed in accordance with this philosophy provides the highest hit rate for such a small predictive cache in rendering connected line segments to a frame buffer.

Another arrangement for implementing a predictive cache in accordance with this invention makes use of a somewhat different caching arrangement than one which surrounds the possible positions to which the next access might be directed. Such an a. angement is less expensive because it uses less hardware but will probably have a somewhat lower hit rate. The arrangement is less expensive in hardware cost because it attempts to predict only a single position for a next access of the frame buffer rather than storing information for all possible positions surrounding a first accessed position.

Such an arrangement may be arrived at by reviewing in more detail the various operations which may occur in a frame buffer. These operations may be broken into specific steps which are illustrated in Figure 4. For example, a series of read operations may occur to determine a line position. In such operations, an address N may be read, then an address N + S where S is some offset from the address N usually within one pixel of the initial pixel. The offset S may be referred to as the stride. Within any particular line segment, the next pixel to be accessed probably lies at N + 2S, whatever the direction of the line segment. The same pixel accesses occur for a sequence of write raster operations. Given the first address and the stride, the next pixels in a sequence of read or write accesses are fairly predictable so long as a single straight line segment is being accessed. It is possible, of course, that the shape of a line segment will vary so much from the few pixels that prediction is not possible; and this fact lowers the hit rate somewhat. For operations involving only writing a line segment to the frame buffer, the same sequences of operations occur as in a read operation.

For operations involving copying pixels of a line segment from one position to another position such as in scrolling and similar operations, more is involved. Memory, including frame buffers, is almost always accessed in amounts larger than a pixel. For example, a typical system accesses an entire word in memory. Such a word may include thirty-two bits or four bytes of memory. In a black and white system in which each pixel is defined by a single bit, thirty-two different pixels are included within each word. Most systems do not allow access of individual pixels; usually a byte (eight pixels in the exemplary system) is the least amount which may be accessed. Thus, a byte or an entire word must be accessed in order to copy even a single pixel from one place to another in the frame buffer. If the byte including the pixel of interest lies at the same offset from the beginning of a word to be read as it does in the word to be written, then a copy operation involves only a read and a write insofar as addresses are concerned. That is, a read at position N is written to position M; a next read at position N + S is written to position M + S. This sequence continues as the line is read and written. Such a read will access and store, in addition to the pixel of interest, adjacent pixels within the same byte or word lying in the same row; if these pixels contain additional line segment positions, they are accessed within the single access of the first position.

However, if the first pixel to be read lies at a first offset from a word boundary in the frame buffer (where additional pixels to be read lie in the same row) and that first pixel is to be written to a position which lies at a second offset from a word boundary, either the portion to be accessed or the portion to be written may lie in two different words or bytes. Copying this information may be

accomplished in one of two ways, either two read operations and one write, or one read and two writes. In one case you need to do enough reads (two) to be able to entirely satisfy a write of an entire word. In the other case you need to do up to two writes to entirely utilize a whole word from each read. In most computers the read/read/write process is used, and this type of copy operation then continues as read/write/read/write. Some copy operations even have a pattern of $R_S/R_D/W_D$, $R_S/R_D/W_D$, etc., the second read being required by the raster operation specified for merging new data (S subscript for source data) with old (D subscript for destination data).

Thus, it can be seen that for any of the typical frame buffer operations, a pattern of read/read/read, write/write/write, or read/write/read/write may be discerned. Knowing this, pattern recognition circuitry may be provided from which the next access can be predicted. For example, if the address of the present information access is desired, its position may be predicted from the last three accesses to the frame buffer. Thus, in general, the last three addresses, the data from the last two accesses, and the type of access (read or write) will predict the next position of a pixel. The type of access defines the pattern as READ/READ/READ, WRITE/WRITE/WRITE, or READ/WRITE/READ/WRITE. Moreover the difference in the addresses predicts the stride for the desired access. As pointed out above, the stride for the READ/READ/READ and WRITE/WRITE/WRITE is the difference between the last and the next to last access addresses. In the case of a copy type of operation in which the pattern is READ/WRITE/READ/WRITE, the stride between the two read accesses is the same as the stride between the write accesses and the stride between a read and a write access is the same as the

last stride between a read and a write access. Consequently, wherever in the pattern for any particular line segment accesses are compared after the first three pixels of a copy or similar operation, the next access is predictable. And even more complex repeating patterns may be made predictable with sufficient logic.

Figure 5 illustrates in block diagram form circuitry 60 which may be utilized to accomplish this more specific form of predictive caching. The circuitry 60 includes registers 61-63 for storing the addresses of each of the last three accesses, the data associated with the last two accesses (usually data is not useful more than two steps back) , and the access type of each of the last three accesses. It is desired to predict and place in the cache the data which is to be accessed at the next step so that an access of the frame buffer is not necessary. The incoming address, data, and operand from the central processor is furnished to a register 64 for comparison with the addresses in the cache.

The addresses of the last three accesses in registers 61-63 are compared in comparators 65-67 to determine if there is a cache hit. For prediction, the address information in these registers is selectively furnished in response to the access pattern determined to subtractors 68 and 69 for determining the stride between read or write accesses depending upon the pattern determined by the comparators. The stride plus the address of one of the last accesses furnishes the predicted address from which information is to be accessed. The information at this address in the frame buffer is accessed

and placed in the cache (77) so that it is available when a new address is received from the central processing unit.

More particularly, if the type of access is found to be of the READ/WRITE/READ/WRITE TYPE, then a copy type operation is predicted. In such a case, the last address A_{n-1} is subtracted from the previous address A_{n-2} to provide the difference between the positions of the read and the write operation. This result is stored in a register 71. The difference between the addresses A_{n-3} and A_{n-1} is then determined in subtractor 69 to provide the difference between the last two read operations or the last two write operations, and the result is placed in a register 72. The results in the two registers 71 and 72 are selected by a multiplexor 75 and added to an access address selected by the multiplexor 76 by an adder 74, and the result is the address predicted. This predicted access's data is placed in the register 77 to be accessed by the central processor. This second, more specialized caching arrangement has a significantly lower miss penalty (one access) than the first arrangement (average of three accesses).

It should be noted that the use of these small specialized caches does not affect in any way the operation of the computer system other than to speed its operation. The predictive cache memory stores information in parallel with the other operations of the system so that the fact that a particular prediction does not turn out to be valid does not delay the system; the operation simply proceeds in its normal manner in the case of a cache miss.

Although the use of predictive caching has been described with respect to frame buffers which have very formal patterns of operation, many of the same characteristics may be found in random access memory. Consequently, the same general principles may be applied to operations within random access memory. For example, there are many situations in which information is stored in or derived from a matrix stored in main memory or is involved in a looping operation. These same principles can be applied to this sort of operation. Memory operations having discernable patterns which may be treated in a similar manner will occur to those skilled in the art.

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.

THE CLAIMS DEFINING THE INVENTION ARE AS FOLLOWS:

1. A predictive caching system for use in computer system having at least one portion of memory in which information is stored for retrieval, and a processor for controlling the access of the memory comprising means for discerning a pattern of access of the memory, means operating in response to the pattern determined by the means for discerning a pattern of access of the memory for determining a next address which will probably be accessed in such memory if the pattern discerned continues, and means for storing the information at the next address determined prior to the next access of the memory whereby the information at the next address is available without the need to access the memory.

2. A predictive caching system for use in computer system as claimed in Claim 1 in which the means for discerning a pattern of access of the memory comprises means for storing information from the last address accessed and the addresses immediately surrounding the last address accessed in the one portion of memory.

3. A predictive caching system for use in computer system as claimed in Claim 1 in which the means for discerning a pattern of access of the memory comprises means for storing information from the last two address accessed and the addresses immediately surrounding the last two addresses accessed in the one portion of memory.

4. A predictive caching system for use in computer system as claimed in Claim 3 in which the means for storing information from the last two address accessed and the addresses immediately surrounding the last two addresses accessed in the one portion of memory comprises a cache memory having storage for information from at least eighteen addresses in the one portion of memory.

5. A predictive caching system for use in computer system as claimed in Claim 1 in which the means for discerning a pattern of access of the memory comprises means for storing information from the last three addresses accessed in the one portion of memory; and in which the means operating in response to the pattern determined by the means for discerning a pattern of access of the memory for determining a next address which will probably accessed in such memory if the pattern discerned continues comprises means for determining a pattern of operations from the information stored regarding the last three addresses accessed, and means for computing a stride between addresses, and means for adding the stride to the last address accessed.

6. A predictive caching system substantially as hereinbefore described with reference to the drawings.

~~7. The steps, features, compositions and compounds~~ disclosed herein or referred to or indicated in the specification and/or claims of this application, individually or collectively, and any and all combinations ~~of any two or more of said steps or features.~~

DATED this FIRST day of MARCH 1991

Sun Microsystems, Inc.

by DAVIES & COLLISON
Patent Attorneys for the applicant(s)

AUSTRALIAN PATENT OFFICE LS

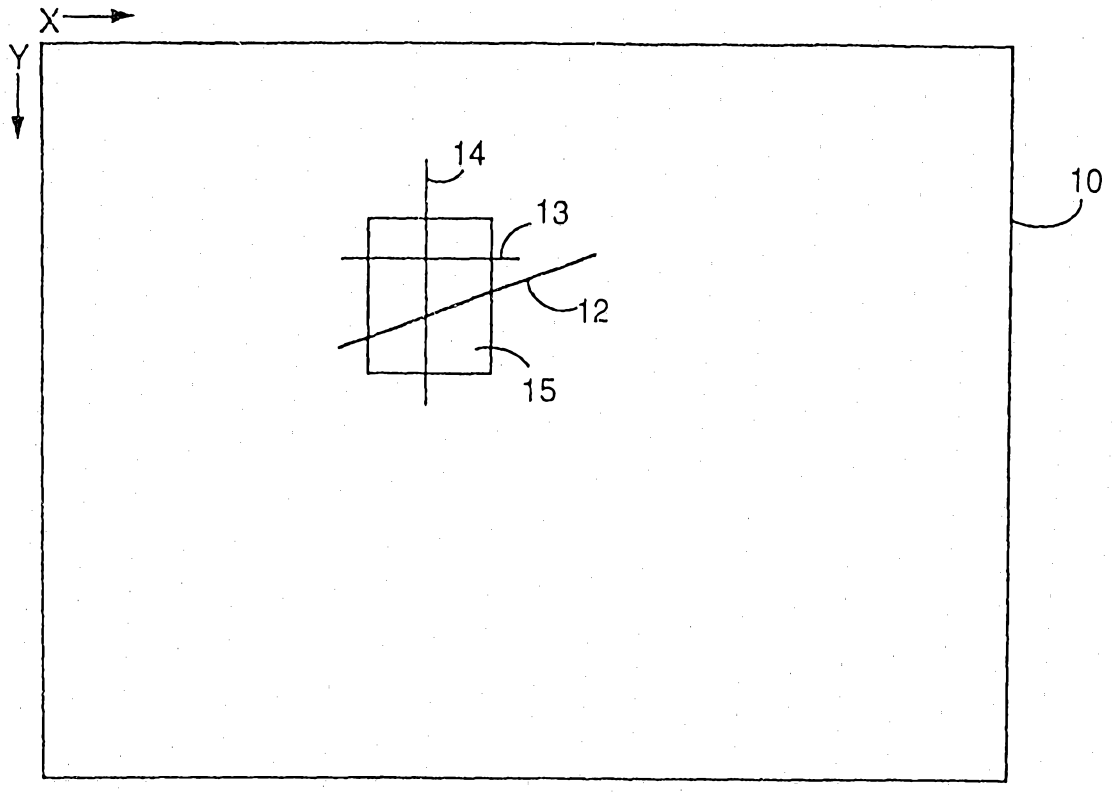


FIGURE 1

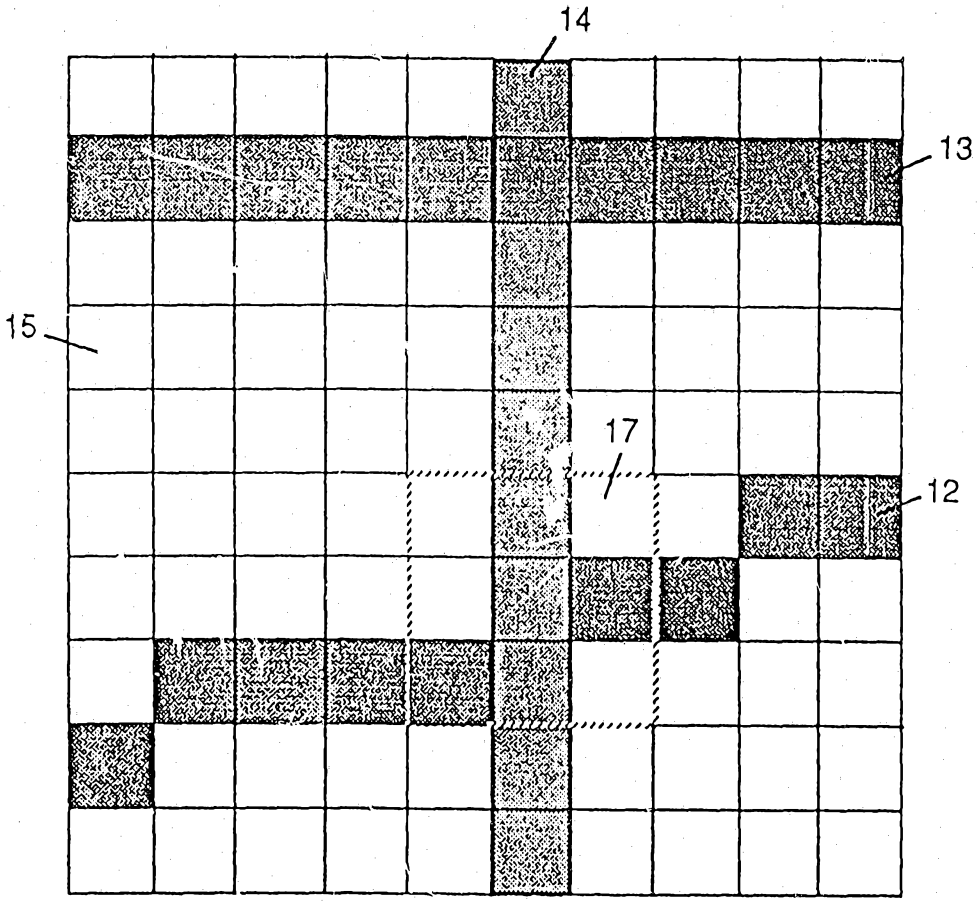


FIGURE 2

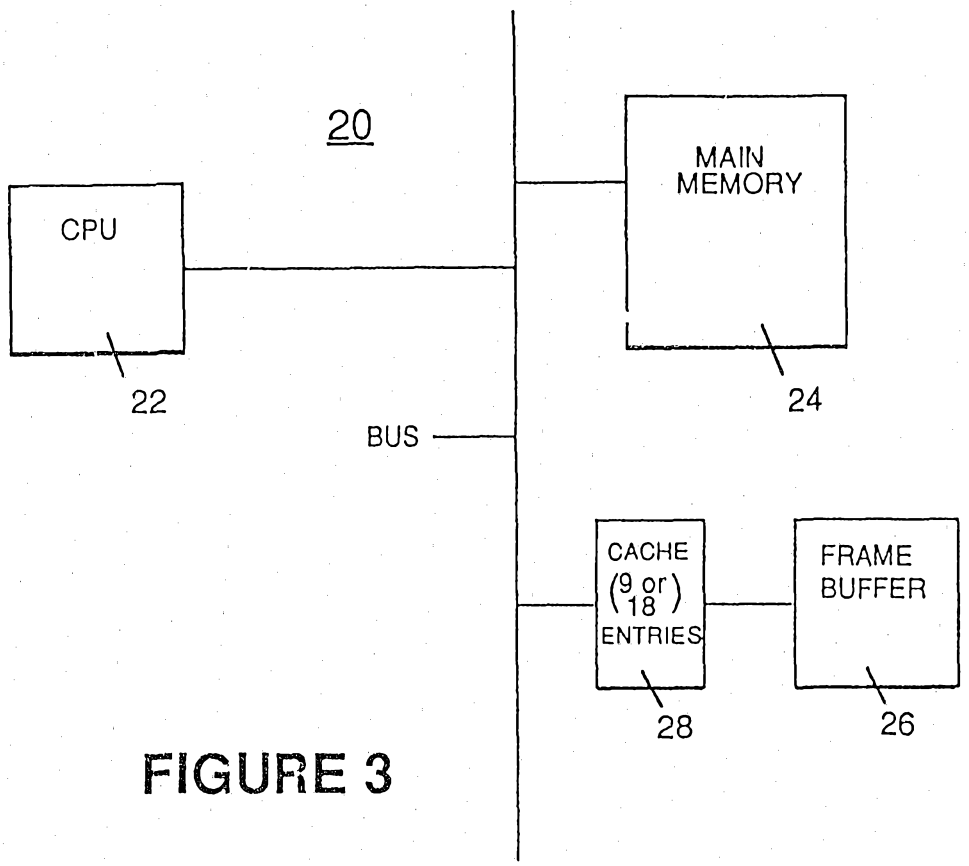


FIGURE 3

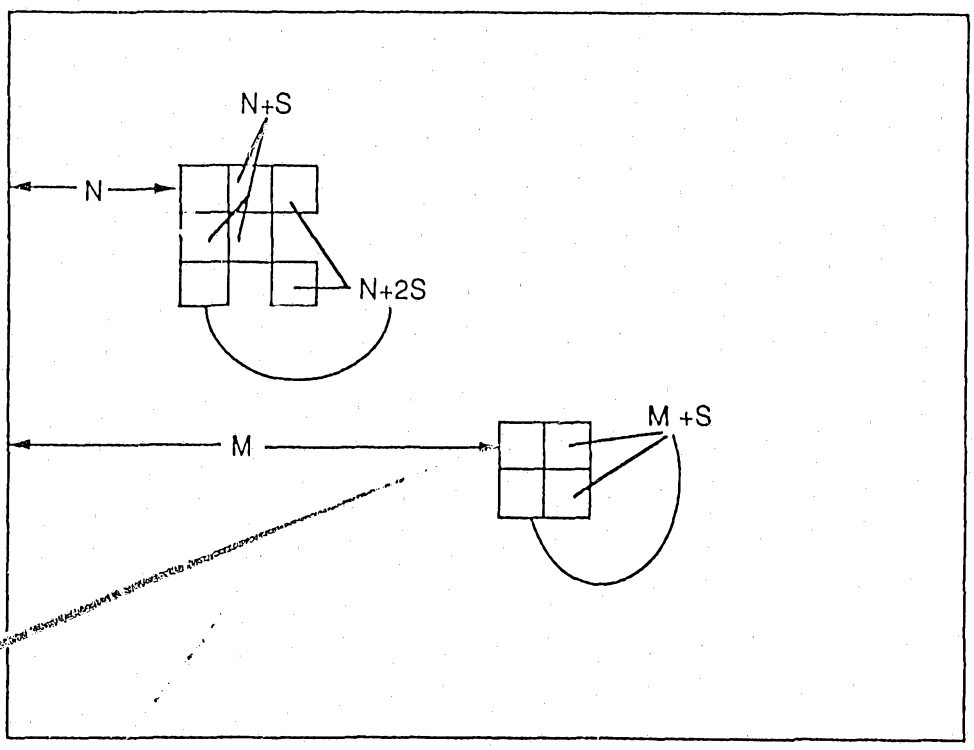
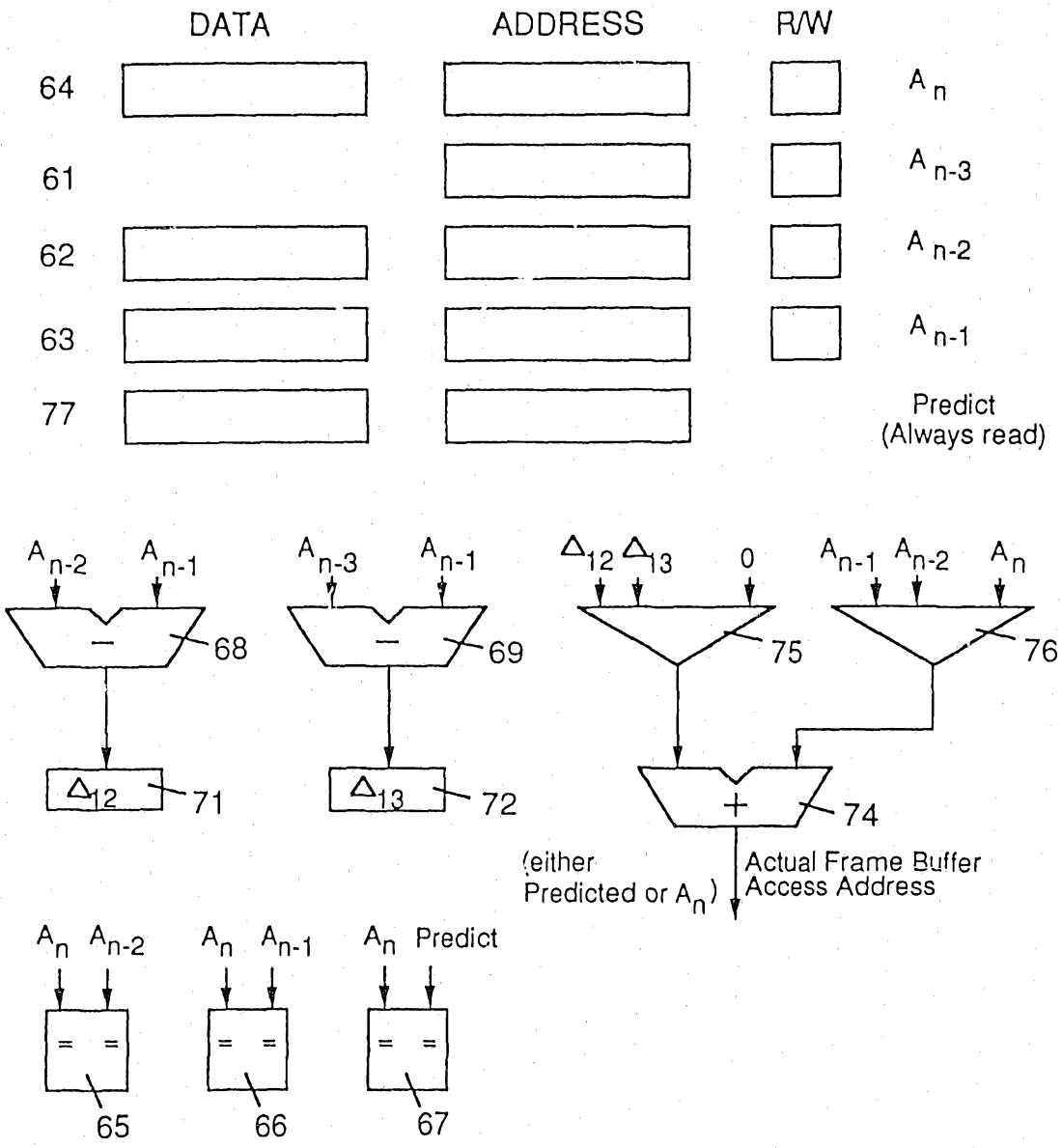


FIGURE 4



RRR patterns: Pred. Address = $\Delta 12 + A_{n-1}$

WWW patterns: Pred. Address = $\Delta 12 + A_{n-1}$

RWRW patterns: Pred. Address = $\Delta 13 + A_{n-2}$

A_n = 1st R of $R_x R_y W_y$ patterns: Pred. Address = $\Delta 13 + A_n$

A_n = 2nd R of $R_x R_y W_y$ patterns: Pred. Address = $\Delta 13 + A_n$

A_n = W of $R_x R_y W_y$ patterns: No prediction necessary

FIGURE 5