

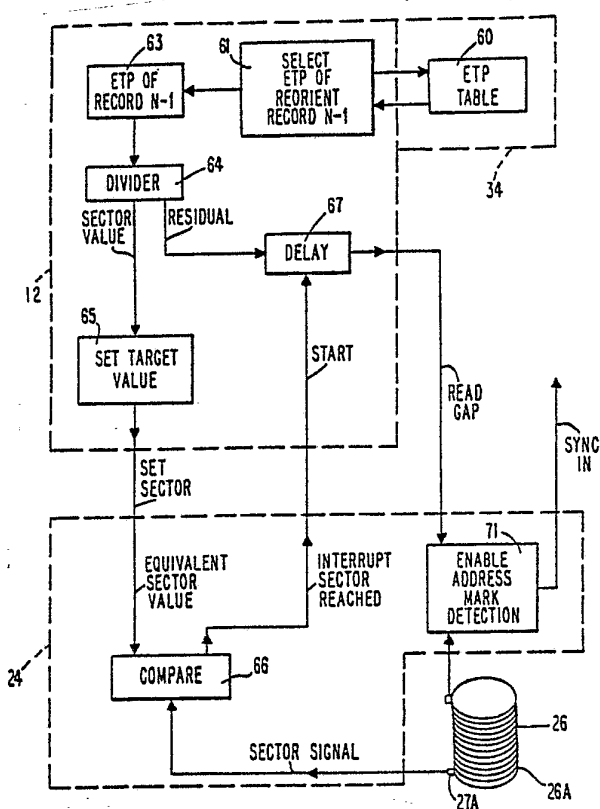


INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification³ : G06F 13/04, 13/06, 3/06	A1	(11) International Publication Number: WO 84/ 02017 (43) International Publication Date: 24 May 1984 (24.05.84)
(21) International Application Number: PCT/US83/01770 (22) International Filing Date: 14 November 1983 (14.11.83) (31) Priority Application Number: 441,818 (32) Priority Date: 15 November 1982 (15.11.82) (33) Priority Country: US (71) Applicant: STORAGE TECHNOLOGY CORPORATION [US/US]; 2270 South 88th Street, Louisville, CO 80028 (US). (72) Inventors: MORENO, Robert, J. ; 5169 Holmes Place, Boulder, CO 80303 (US). TREDE, Brian, E. ; 1215 Delphi Drive, Lafayette, CO 80026 (US). SCHMIDT, Robert, W. ; 1245 Doric Drive, Lafayette, CO 80026 (US). (74) Agents: KURTZ, Richard, E. et al.; Woodcock Washburn Kurtz Mackiewicz & Norris, 1800 United Engineers Bldg., 30 S. 17th Street, Philadelphia, PA 19103 (US).		(81) Designated States: DE, GB, JP. Published <i>With international search report.</i>

(54) Title: CONTROL OF CACHE BUFFER FOR MEMORY SUBSYSTEM**(57) Abstract**

A solid-state cache memory subsystem configured to be used in conjunction with disk drives (26) for prestaging of data in advance of its being called for by a host computer (10). Buffer controller (32) is used for establishing and maintaining precise correspondence between storage locations in the solid-state array (34) and on the disk memory. The correspondence is used for establishing a reoriented position on a disk in the event of error detection, and in order to determine when a predetermined quantity of data has been read from the disk into the cache in a staging operation.



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.

AT	Austria	LI	Liechtenstein
AU	Australia	LK	Sri Lanka
BE	Belgium	LU	Luxembourg
BR	Brazil	MC	Monaco
CF	Central African Republic	MG	Madagascar
CG	Congo	MR	Mauritania
CH	Switzerland	MW	Malawi
CM	Cameroon	NL	Netherlands
DE	Germany, Federal Republic of	NO	Norway
DK	Denmark	RO	Romania
FI	Finland	SE	Sweden
FR	France	SN	Senegal
GA	Gabon	SU	Soviet Union
GB	United Kingdom	TD	Chad
HU	Hungary	TG	Togo
JP	Japan	US	United States of America
KP	Democratic People's Republic of Korea		

CONTROL OF CACHE BUFFER FOR MEMORY SUBSYSTEM

Cross Reference to Related Applications

This application is a continuation-in-part of copending Serial No. 325,288 filed November 27, 1981, Moreno et al, having the same title.

5 Field of the Invention

This invention relates to improvements in cache memory subsystems used in data processing systems to improve the overall throughput or utility of the data processing system. In particular, the invention relates to organization of the
10 cache buffer memory as to header and data formatting, and to control of the correspondence between the cache buffered data and its home position on magnetic storage media.

Background and Objects of the Invention

Improvements in data processing systems have generally
15 been directed at reduction either of the average time required to execute a given instruction or reduction in cost of the equipment required to perform such an instruction. One design tradeoff which has typically been made is that of cost versus speed for units of memory for the storage of data. For
20 example, tape memory is traditionally slower and less expensive than disk memory. Disk memory in turn is available in several types; the selection of any one type involves a cost/speed tradeoff. Disk memory is slower but less expensive than solid-state memory which itself is available in several types,
25 the selection of which again involves a cost/speed tradeoff. Thus, it continues to be a need of the art to provide cheaper, faster memories or, failing that, to improve the efficiency of presently existing memory types. The present invention relates to an improvement of the second type. In particular, the



- 2 -

invention involves apparatus and methods of operation thereof for reducing the average time necessary for a host central processing unit (CPU), which typically comprises an arithmetic and logic unit and a main memory unit for retention of the instructions and data currently being operated on, to obtain data stored on a less expensive, long-term data storage device, such as a magnetic disk or tape drive unit.

Delays in memory access occur due to mechanical limitations on the apparatus. For example, in the case of a disk drive, in general, plural disks rotate at a fixed speed past read/write heads which may either be stationary with respect to the disk or move radially back and forth with respect to the disk in order to juxtapose the heads to various portions of the disk surfaces. In either case, there is a finite average time required for a particular data record to be located and read from the disk into a faster form of memory, typically a solid-state main memory included in the host computer. The delay may involve the "seek" time required for the head to be moved radially to the particular "track" selected, as well as "latency" time required for the disk to rotate with respect to the head until the beginning of the particular record sought is juxtaposed to the head for reading and writing.

Accordingly, it is an object of the invention of copending application Serial No. 325,346 filed November 27, 1981, of Dodd entitled "Cache Buffered Memory Subsystem", incorporated herein by reference to provide a memory subsystem in which the average time required for a record sought to be transferred to the main memory system of a host computer is significantly reduced. The present invention relates to details of a preferred embodiment of that invention.

Prior art data processing systems typically comprise a host computer and long-term memory storage means including such devices as magnetic disk memory units and magnetic tape units. Communication from the host computer to the disk or



- 3 -

tape memory subsystem is generally made via a "channel" which physically comprises a defined set of signal connections over which all information--including data as well as commands, control signals, status signals, request signals and the like-- must pass. In order that a memory subsystem can be marketable, it must "interface" or mate directly with a channel identical with those with which prior memory subsystems mated, thus being "plug compatible" with the host computer.

It is an object of the present invention to provide an improved memory subsystem which is "plug-compatible" with a prior art host computer.

Similarly, it is desirable if not commercially requisite that any data subsystem not require modification to the host computer's programming instructions or "software" upon connection, i.e., that it be "software-transparent" to the host.

It is an object of the present invention to provide a memory system which provides improved performance as noted above, while being software-transparent to the host.

The copending application referred to above and incorporated by reference herein refers to a cache buffer memory subsystem comprising a solid-state cache memory and a cache memory controller. The solid-state cache memory contains data written therein in advance of a host's request for that data. Thereafter, when the host calls for that data it can be supplied to the host immediately inasmuch as there is no seek or latency time required to access any portion of a solid-state memory array. In a preferred embodiment of that invention, described more fully in copending application Serial No. 325,350 filed November 27, 1981, of Dodd et al, entitled "Detection of Sequential Data Stream", and a continuation-in part of that application, Serial No. _____ filed _____ (STC-159), both incorporated herein by reference, the cache memory controller determines at what points it would be desirable to "stage" data not having



- 4 -

been called for by the host into the cache memory array in anticipation of its being called for.

Those skilled in the art will recognize that disk memories of the type in most common use at the present time generally comprise a plurality of disks each having upper and lower surfaces for the storage of data by magnetization of areas on their surfaces. The disks are divided into concentric tracks which are divided into sectors. An index mark is provided on each disk identifying a first sector. When the host computer desires data from a disk drive it issues a "SET SECTOR" command whereby the disk drive is notified that data beginning with the sector identified by the host is to be read. According to a preferred embodiment of the present invention the data called for in each reading operation initiated by the host is examined to consider whether it is likely to be a part of a sequence of such records to be called for. If so, it is considered that the following record is likely to be called for, and the entire track from which the access request was satisfied is cached. If the remainder of the track is then called for by the host, it is available in cache memory and the host's request can be very quickly satisfied. Moreover, if the entire track is thereafter called for, the succeeding track is then "prestaged" to the cache in anticipation of further requests.

Those skilled in the art will recognize that numerous types of disk drives are available, each having differing amount of data stored per track. Accordingly, if the cache memory subsystem is to store data by tracks efficiently, it must be adapted to contain varying lengths of data efficiently. This is satisfied in accordance with copending application Serial No. 325,351 filed November 27, 1981, of Coulson et al, entitled "Adaptive Domain Partitioning of Cache Memory Space", and a continuation-in-part of that application, Serial No. _____ filed _____, having the same title (STC-166).



- 5 -

While, as noted above, it is deemed desirable to bring data into the cache from a disk memory system track by track, it is not necessarily the case that each request will begin at the index mark which may be taken to the start of each track. Instead, if the host has called for the first record in the track, the subsequent record will inevitably start other than at the beginning of the track. Clearly, it would be desirable to read succeeding records into the cache without waiting for the disk drive to complete the particular rotation during which the first data record was read. In order to do so, it is clearly essential that accurate orientation of the relative location of the track on disk and its storage location in the cache be maintained so that, for example, host requests for data stored on the remainder of the track can be satisfied by access to the corresponding data storage addresses in the cache.

It is an object of the present invention to provide such accurate orientation of disk memory with cache memory.

Those skilled in the art will recognize that records when stored on disk drives have "headers" or count fields preceding each record which include information identifying the particular record, specifying its length and the like. As is understood in the art, the requirements of headers for data stored in solid-state memories are substantially different from those of count fields on magnetic disk media, such as disk drives, and it would be accordingly desirable to provide means for transforming a number of magnetic media count fields into a single solid-state memory header.

It is an object of the invention to perform this transformation of the count fields into a single header and, in particular, to provide a header of a length variable in accordance with the number of records on each track and not of fixed length as is usual in the magnetic media prior art.



- 6 -

As noted above, it is desirable that an entire disk track be read from disk memory into cache memory at one time, but not necessarily starting from the beginning or index mark of the track. Accordingly, it is an object of the present invention to provide a means whereby it can be reliably determined that an entire track of data has been read from a disk drive into the cache memory array.

Occasionally, it is necessary to convert a channel program that was operating on a track image in cache memory (a frame) to operating on the actual track on the disk; that is, it becomes necessary to read a portion of a track from the disk, rather than the cache, or to write to a track which has been cached. In each case, it is necessary to recreate the specific record and field orientation that existed in cache before continuing the channel program with records from the disk track.

The need for returning a channel program to disk arises in the following situations: 1) an error prevents data from being accessed in cache memory; 2) a write command is received in the channel program and it is deemed undesirable to write to cache memory; or 3) a sequential track required during multi-track processing is not located in cache memory.

It is an object of the present invention to perform an accurate reorientation from the records stored in cache memory to the same records stored on the disk track. Further, it is an object of this invention to perform this reorientation operation within a single revolution of the disk, without necessarily waiting for the index mark on the disk track to provide reorientation.

In the prior art, this was not always possible. Examples of related prior art systems include the Storage Technology Corporation 4000/4305 "solid-state disk" system, in which a solid-state memory effectively minimized a magnetic disk system and the Storage Technology Corporation 8880/8650 disk subsystem.



- 7 -

The 4000/4305 solid-state disk subsystem used a track remaining count which is similar to the emulated track position (ETP) of this invention. However, this track remaining count was not used for transferring orientation from a frame to a disk track; no magnetic disk memory was included in that system.

The 8880/8650 disk subsystem uses a reorient counter that counts an approximate full track revolution starting from the beginning of a record. If the 8880 director encounters errors reading a record, then the reorient counter in the device informs the director when the same record is about to pass under the read/write head again. This technique allows positive reorientation on a specific record on disk, but requires that orientation had been previously established on the disk.

Summary of the Invention

The above mentioned needs of the art and objects of the invention are satisfied by the present invention in which reliable orientation of the cache memory with respect to the disk drive is maintained by generating a table of "emulated track position" (ETP) values equal to the approximate addresses on disk of the records being stored in the cache. The first ETP is the starting address which corresponds to the beginning address of the first record stored in the cache. The first ETP is thus a measure of the number of bytes from the index mark on the disk at which the starting point of the cached data was located. This value is saved as a "target" ETP, and is used to determine when a full track of data has been read. The next record's count field information is read and saved in a second register. The records on the track between the current orientation point and the index point can then be read. When the disk's index mark is reached, a current ETP value is initialized to zero in a third register. The data records are read from the index into the cache. As each field is read, the



- 8 -

current ETP value is updated to count the number of bytes that have passed. After each incrementation the current ETP is compared to the target ETP. When it is greater than or equal thereto, the count field information from the last record read is compared against that stored in the storage director. If they are not equal, record reading continues. If they are, an indication is provided that the entire track has been read.

In accordance with the present invention, the ETP values can be used to reliably retransfer orientation of data being read from the cache back to the disk drive in the event of an error occurring in the middle of a read operation, or in the event of other conditions previously discussed which require returning the channel program to disk. In carrying out the invention, the ETP value for the record preceding the record upon which reorientation is to be obtained, i.e., the first record called for by the channel, is selected from the table of ETP values. The selected ETP value is converted into an equivalent sector number and a residual value by dividing the emulated track position by the number of bytes per sector, which is fixed. This produces an equivalent sector value which is supplied to the disk drive control module, where it is compared to the sector signal produced by the disk drive representing the sector currently rotating under the heads. When the desired sector is juxtaposed under the heads, the control module produces an interrupt signal which results in the director enabling the reading of records from disk after a further delay. The length of this further delay is controlled by the residual value, to align the beginning of reading with the correct record in the sector previously identified. The director enables the reading of address marks during a time period or "window" at the termination of the delay. In this manner, accurate detection of the desired record count field is obtained. Reliable and accurate reorientation is thus performed.



- 9 -

The foregoing and other objects, features and advantages of the invention will be better understood from the following more detailed description and appended claims.

Brief Description of the Drawings

5 The invention will be better understood if reference is made to the accompanying drawings, in which:

Fig. 1 shows a schematic overview of the system of the invention;

10 Fig. 2 shows the data and control paths in a simplified view of the system of the invention;

Fig. 3 shows an expanded view of connections between the directors, the file interface unit, and the cache buffer and manager units;

15 Figs. 4A and 4B depict the cache manager operations during a stage to cache and during reorientations;

Fig. 5 shows a sample header format;

Fig. 6 depicts the header, home address, count and data fields of a disk track which has been stored in cache memory; and

20 Figs. 7A - 7C are timing diagrams respectively showing the home address, count and data fields as read from a disk track, the track sector signal, and the sector value signal.

Description of the Preferred Embodiments

25 As discussed above, the cache buffer memory subsystem of the present invention is designed to operate in accordance with a prior art host computer in such a manner that no modification of the host's hardware or software is required. In particular, the cache buffer memory subsystem of the invention should attach to the host computer through a
30 conventional memory channel without modification; that is, it should be plug-compatible with, and software-transparent to, the host. In a preferred embodiment, the cache memory subsystem of the invention is designed to operate in accordance



- 10 -

with an IBM Corporation computer (or other computer plug-compatible with the IBM unit) of the class in which the channel is defined by IBM Publication GA22-6974-4 entitled, "I/O Interface Channel to Control Unit". This volume describes the channel specifications for plug-compatible memory systems. In a particularly preferred embodiment, the cache buffered memory subsystem of the invention is comprised in a disk drive memory system of the type previously sold by the assignee of the present invention, reconfigured to include a solid-state random access memory (RAM) which can physically be included within the enclosure containing the disk drive storage director unit. Indeed, the addition of the cache feature may be made in such a manner as to be field-installable on previously shipped customer units at minimal additional complexity and cost, while substantial performance improvements may be realized.

Referring now to Fig. 1, an overall view of a data processing system comprising a host computer system as defined above and a cache buffered memory subsystem is shown. The host computer 10 is connected to a pair of storage directors 12 and 14 through channels 16 and 18, respectively. The directors and channels can be multiplied further, as well known in the art. Each director comprises a file interface 20 through which data is passed, via control modules indicated generally at 24, and thence to disk drive units indicated generally at 26. As indicated schematically in the drawing, the disk drives 26 may be of various types. The control modules 24 to which the disk drives 26 are connected serve to interface varying types of disk drives with directors 12 and 14 such that plural types of disk drives 26 together with appropriate control modules 24 can be used in conjunction with identical directors. As described thus far, the system of Fig. 1 is conventional. According to the present invention, the directors 12 and 14 are modified to additionally comprise cache interface units 28 which serve to interface the directors 12 and 14 with a cache buffer memory subsystem according to the invention, which as schematically



- 11 -

shown comprises a buffer unit 30 and a buffer control unit or manager 32. The buffer unit 30 has as its chief function the storage of data having been "staged" in anticipation of being required by the host computer. When the staged data is called for it can be very quickly read under control of a cache bus controller 35 out of a solid-state random access memory (RAM) array 34, the heart of the buffer unit, through one of a plurality of cache port controllers 36 and 38, respectively communicating with the directors 12 and 14, and thence to the host. This is generally a considerably faster process than reading the data directly from a disk drive unit 26 as there is no latency time involved in the use of the random access memory 34, as there is in the case of the disk memories 26.

Referring now to Fig. 2 a schematic view is shown of the data flow between the file interface 20 and the host interface 16 in the cache buffered memory subsystem. The typical prior art data path directly from the channel interface 16 through a director 12 and via a file interface 20 to a disk drive, sometimes termed "DASD" for "direct access storage device" 24 is shown. This data path is preserved and is used, as will be discussed below, for all ordinary read operations, when there is an error in the cache buffer, and during ordinary write operations. In a staged read operation, however, data is read from DASD 24, passes from the file interface 20 through the director 12 and thence to the cache buffer 30, under the control of the buffer manager unit 32. When the host 10 then calls for the data actually to be supplied to it, the data passes out of the buffer 30 through the director 12 and through the channel interface 16 to the host 10. It will be appreciated that parallel connection is provided between the two directors 12, the channels 16, the buffer 30 and the file interface 20. That is to say, data may pass from a first director 12 from the buffer 30 enroute to the channel interface 16 after having been passed from DASD 24 to buffer 30 for staging by a second director 12. Such "paralleling" of



- 12 -

directors 12 also permits data to move from the cache 30 to the host 10 even if the director 12 used to stage data into the cache buffer 30 then takes up another task before the data is actually called for by the host 10.

5 In this connection, it will be appreciated that the present IBM channel design provides a sequence of commands e.g., before an actual readout operation takes place. Typically, the host will issue a "START I/O" command to the channel." The channel in processing the "channel program" will
10 issue a "SEEK" command to the director which will then cause the disk drive's head moving apparatus to access the proper track of the disk. When this has been done a signal is passed back to the channel indicating that the head is in position. After receipt of the "READ" command, the disk drive examines
15 the count and key data fields for that containing the record of interest; when it is juxtaposed to the read/write head, the record is read via the director to the host's main memory for processing. According to the present invention, the same practice is followed upon receiving of a first read command.
20 However, the channel program is also examined by the cache manager 32 for indications that the successive record is likely also to be called for by the host. If so, the buffer manager 32 then causes the director to have the succeeding record or records stored on the disk copied into into the cache buffer 30
25 so that subsequent read commands for successive records can more quickly be complied with. As noted above, the determination of which records appear to be members of a sequence of records to be called for by the host is the subject of copending application Serial No. 325,350 filed November 27,
30 1981, of Dodd et al entitled "Detection of Sequential Data Streams", and a continuation-in-part of that application, Serial No. _____, filed _____, having the same title (Attorney's Docket No. STC-159). In a particularly preferred embodiment, data is staged from the disk into the
35 buffer an entire track at a time. Such disk tracks typically



- 13 -

may comprise up to on the order of 100 records, i.e., blocks of data separated from one another by gaps and identified by headers comprising index marks and the like. The 100 records may in turn be part of a disk file which may occupy many tracks, or in the case of large data files, up to several complete disk units.

As noted above, it is desirable that the cache buffered memory subsystem be a field installable modification to a pre-existing product thus adding minor additional cost and complexity to a memory subsystem while permitting substantial performance advantages.

Fig. 3 shows one way in which the cache buffer feature of the invention can be added to a preexisting product while satisfying these conditions. Fig. 3 shows four directors 12 and 14 each connected to a host computer via a channel 16. As in the prior art the directors 12 and 14 are each connected by way of the file interfaces 20 to the control modules (not shown) which, for example, convert the director commands to forms suitable for accessing a particular type of disk and so forth. Thus, the control modules are selected in accordance with the type of disk storage media to which they are to be connected, while the directors 12 and 14 are the same regardless of the type of disks used. Such directors and control modules may be as presently found in commercial products of the assignee, for example, in the Storage Technology Corporation Model 8880 disk controller and Model 8650 disk drive units respectively. According to the present invention, to each director 12,14 is added a cache interface unit 28 which is connected to a buffer interface 36, connected to the bus controller 35 of the cache buffer unit 30.

Control lines are also connected to the cache interface lines which run to the control interfaces 42 of the buffer manager 32. Thus, to the preexisting Model 8880 unit one need only add cache interface cards to the directors, and install the cache buffer and buffer manager units. Such an installation can be



- 14 -

made to preexisting units in the field within a reasonable period of time and at reasonably modest additional cost.

It will be appreciated from inspection of Fig. 3 that the cache buffer 30 is connected in parallel to each of the directors 12,14 as is the buffer manager 32 which controls flow of data between the directors and the cache buffer. This is in distinction to, for example, a cache arrangement proposed by Memorex Corporation in which a single cache per each director is interposed between the control module and the director. This arrangement has several disadvantages; one of the prime disadvantages is that more caches are required. While for a given performance level such single director caches can be made somewhat smaller, they are then less likely to be fully utilized. Moreover, provision of a separate cache for each director does not allow two or more directors simultaneous access to the cache which is desirable. This alternate path access is very desirable in a highly active system to avoid path busy conditions. Such versatility is made possible by the arrangement of the invention as illustrated in Fig. 3.

As discussed above, in a preferred embodiment of the invention, it is configured as an add-on subsystem to a previously existing product, namely the Model 8830 disk controller manufactured by the assignee of the present invention, Storage Technology Corporation. In this embodiment, the director 12 performs the functions of the director in the Model 8880; that is, it directs and interfaces to the control modules (which, for example, demultiplex the data to serial form, for writing it to a disk) thus directly interfacing the drives themselves to the host channels by converting the data from the form in which it is stored on disk to one to which the channel can respond. The director 12 according to the invention performs the additional functions of buffered data transfer, and execution of the stage operation, i.e., writing of data from the disk 26, (Fig. 1) via the control module 24 into the cache buffer 30. The director 12 also provides buffer



- 15 -

error recovery systems, which amount to bypassing of the buffer in favor of direct transfer of data from the disk 26 to channel interface 16, and the director communicates with the buffer manager 32. This is to say, the director 12 performs switching functions, controlling the flow of data between the file interface 20, to which the disk units 26 are connected, the cache buffer 30 and the channel interface 16 to which is connected the host computer. These switching functions are controlled in accordance with instructions received from the buffer manager 32. Thus, in the preferred embodiment in which the memory subsystem of the invention is added onto a preexisting STC 8880 product, the buffer manager 32, the cache buffer 30 and suitable control and data lines are added, so that the director 12 controls the data path in accordance with the instructions of the buffer manager 32.

As noted above, the buffer manager 32 determines whether a particular data record or series of records is likely to be called for by the host and accordingly decides whether or not to "stage" it into the buffer 30 in advance of its actually being called for by the host in order to save access time. The specific decision-making process performed by the buffer manager 32 in making this determination is the subject of copending application Serial No. 325,350 filed November 27, 1981 of Dodd et al entitled, "Detection of Sequential Data Stream", and of a continuation-in-part of that application Serial No. _____ filed _____ (STC-159). Broadly, the decision involves examining the channel program data to look for track jump orders, end-of-file indicators, write indicators and the like, events unlikely to be encountered during reading of sequential data sets.

The cache buffer 30, which performs the function of storing the data, is preferably organized by frames. In a preferred embodiment, the cache is subdivided into "domains" of a fixed size chosen such that the individual frames which correspond to logical tracks on disk fit as closely to



- 16 -

integrally as possible within the domains. This cache space organization scheme is the subject of copending application Serial No. 325,351 filed November 27, 1981, of Coulson et al, entitled "Adaptive Domain Partitioning of Cache Memory Space", and a continuation-in-part of that application, Serial No. _____ filed _____ (Attorney's Docket No. STC-166) both incorporated herein by reference. The cache buffer 30 also stores the control information or header which defines each individual variable length logical frame by count, key and data fields as noted above.

The buffer manager 32 in the preferred embodiment comprises a microprocessor 40 (Fig. 1) and performs the management function as follows. The chief function of the buffer manager 32 is to manage the space of the cache, that is, to direct the storage of data within the cache memory 30 and to retain the locations at which the specific tracks are stored in a "frame directory." The buffer manager 32 similarly schedules stage operations and communicates its instructions to the director 12. The buffer manager 32 also performs error management and recovery functions, which include switching the flow of data directly from the disk to the channel interface in the case of a catastrophic error, which is the subject of copending Serial No. 325,287 filed November 27, 1981 of Blickenstaff et al, entitled "Data Path Reconfiguration for Error Control", incorporated herein by reference.

The present invention is an improvement on the cache memory subsystem described above. After a first record has been determined to be part of a sequence of records, the remainder of the disk track is then desirably cached. This presents problems in address control and in correlation of the cached data with the data received or written on disk. In particular, inasmuch as the host's subsequent commands will be in terms of disk sector addresses and record numbers, not cache storage locations, accurate and fool-proof means must be provided to insure that the proper correlation is maintained.



- 17 -

Moreover, it is also necessary to provide means for determining when a full track has been staged into the cache so that the staging operation can be terminated at precisely the correct time. The present invention solves these problems.

5 It would be a simple matter, of course, to begin staging at the index mark which is recorded at one location on each disk track and continue until the index mark was once again encountered. However, ordinarily the first record on the track will be that which is first accessed by the host, and
10 hence most of the track will not have been read to the host at the time the determination to stage the remainder of the track is made. Therefore, the time taken for the disk to rotate the remainder of a revolution would be consumed while waiting for the index mark once again to be juxtaposed to the read/write
15 head. This waste of time is clearly undesirable.

Additional constraints are posed by the requirement that varying sizes of disk tracks be supported by the cache memory according to the invention. This prevents the cache memory from being divided into subportions or "frames" of sizes
20 equal to a particular disk track size and then simply assigning the starting addresses of each of the frames to the contents of the disk track as they are staged. In the preferred embodiment, the cache memory subsystem of the invention is instead divided into domains which are sized to contain
25 substantially integral numbers of frames of varying sizes to accommodate varying disk track sizes. The domains themselves can be reallocated to other types of frames when desired. This is the subject of the copending Serial No. 325,351 filed November 27, 1981, of Coulson et al entitled "Adaptive Domain
30 Partitioning of Cache Memory Space" and a continuation-in-part of that application, Serial No. _____, filed _____, (Attorney's Docket No. STC-166), both referred to above.

Fig. 4A will be referred to here to explain the way in which correspondence between the place on the disk at which a
35 track of data to be stored in the cache is located and its



- 18 -

address in the cache is maintained. A disk pack 26 is shown in Fig. 4A. The disk pack may comprise a plurality of data storage disks 26B which are accessed by read/write heads 27 and a dedicated disk 26A which contains permanently encoded servo and sector identification information. This information is read by a dedicated head 27A. As can be seen from Fig. 4A and the enlarged portion thereof shown above, each data surface 26B is divided into a plurality of sectors 77. Circumferentially extending tracks 72 extend across the sector boundaries 77A. As is usual, each track may be divided into one or more records indicated as R1, R2, etc., which each in turn comprise a "header", i.e., count field, indicated at H and data indicated by D. As shown, the record boundaries do not necessarily coincide with the sector boundaries 77A. The count field information contains, for example, control information such as the length of the following data record, synchronization and address marks and is used to identify the specific record within the sector. Accordingly, in order to locate a particular record on disk, one must know the disk surface, the track number, the sector number, and the record number, the last of which is to be found within the count field. The first sector on the disk is located by an index mark 73 permanently written to the servo disk 26A from which sector numbers may be counted.

It will be apparent to those skilled in the art that if it is desired to cache a track of data from a disk to a cache memory, and if a portion of that track has already been read to the host computer, it would be preferable to begin caching with the remainder of the track, rather than wait for the index mark 73 to once again be juxtaposed to the read/write head 27. Accordingly, when it is desired to stage a track to cache, a frame of the appropriate size is selected from the list of available frames of that size and beginning and ending addresses in the cache memory array 34 are then calculated by the cache manager 32 in accordance with the invention of



- 19 -

compending application Serial No. _____
filed _____, of Swiatek et al, entitled "Dynamic
Addressing For Variable Track Length Cache Memory", (Attorney's
Docket No. STC-155) incorporated herein by reference. Data is
5 then read from the disk to the cache beginning at a starting
address. In order that correspondence can be maintained
between the data read from the disk (which, as noted, does not
begin with the index mark) the current sector address must be
stored. The current sector number can be read from the
10 dedicated disk surface 26A using the information coded thereon,
as well understood in the prior art. According to the present
invention, the sector number is converted to an emulated track
position (ETP) within the cache director 12, at 50, and this is
stored as the target ETP in a target ETP register 51. This
15 initial ETP is a measure of the number of bytes from the index
mark at which staging of data from the disk to the cache 34 was
begun. A current ETP register 51 is initialized to zero and is
used in building a table of ETP values at 60, each entry
representing the relative ETP for a record. The record number
20 within the first count field found is stored in a target record
number register 52, also comprised within the cache director
12. Thereafter, reading of the records on the track between
the starting position and the index mark proceeds, whereby the
data is read into the cache array complete with the count
25 fields. As each record is staged from DASD to cache, the
current ETP value is incremented and an entry is made into the
ETP table at 60, giving the beginning address for that record.
When the index mark 73 is reached, a current ETP register 54 is
set to a value equal to the number of bytes between the index
30 mark and the home address field. The remainder of the records
between the index mark and the point at which caching of the
data began are then read from the disk to the cache array 34.
As each subsequent record is read, the current ETP value in
register 54 is incremented to reflect the number of bytes which
35 have passed. Whenever the contents of current ETP register 54



- 20 -

are thus updated by the passage of another record, its value is compared at 53 with that stored in the target ETP register 51 to determine whether the first record staged is once again approaching the read/write head of the disk. When the current
5 ETP is greater than or equal to the target ETP, data has been read from each sector of an entire track of data. When this occurs, the record number (abbreviated REC NO on the drawing) from the current record as comprised in a current record number register 55, to which the record numbers are copied as they
10 pass by the head, is compared at 57 with the record number in the target record number register 52, written to previously as discussed above. When the target record number 52 equals the record number of the current record stored in current record number register 55, it may be concluded that all records in the
15 sector in which staging was started have been read, and accordingly an indication may be provided as at 58 that a full track has been copied to the cache. The entries in the ETP table 60 for records staged prior to encountering the index mark are now adjusted from the relative values originally
20 stored in the ETP table 60 to their actual values. This is required because the exact ETP value for the first record staged is only known after having read all records between index and this first record. The ETP table 60 may then be copied from the director 12, where it is generated, to the
25 cache array 34, where it is stored together with the data, as discussed further below.

In order to use the "wraparound" method of writing data from disk into cache, whereby the operation can begin wherever convenient, rather than waiting for the index mark to
30 be juxtaposed to the read/write head, the header information written at the beginning of each frame in the cache must comprise means to indicate the relative locations of the records within the frame, so that, for example, when the host later calls for "record 3", there is a directory of cache
35 addresses showing where this record is stored in the cache frame. For example, suppose that there are six records on a



- 21 -

given track on a disk drive. The host calls for the first record. It is read directly from the disk to the host. Examination of the channel program by the buffer manager in accordance with the copending applications of Dodd et al referred to above reveals that the subsequent records are likely to be called for as well. This determination consumes some rotational time, so that records 3 through 6, beginning with record 3, are read into the cache beginning at a stated beginning address (BOT). When the index mark is reached, the cache director 12 initializes the present emulated track position (ETP) register 54 and proceeds to read the remainder of the track into the cache, ending with record 2. The host will thereafter presumably call for records 2, 3, 4, 5 and 6, though not necessarily at once. In this connection note that the cache memory subsystem's design is again governed by the requirement of software-transparency. Here this means that the memory subsystem cannot be designed to respond to a request for the "next" record, it must be capable of responding as if to unrelated commands. Accordingly, later the host may call for, e.g., record 5. The cache director then consults its directory, locates the frame and examines the header, so as to be able to index a cache memory address pointer directly to the beginning of record 5, i.e., without reading through records 2 through 4, which is important if the cache is to operate efficiently.

Fig. 5 shows how the header which appears at the beginning of each frame can be organized to provide this wrap around capability. Stated slightly differently, because of the nature of the stage procedure, the records are not written into the cache frame assigned starting with a home address (HA) field. Instead, the stage operation begins with the first available count field in order to minimize the time spent staging. Since the frame header is built as the records are being staged, the header entries will also start with the first available count field instead of the HA entry. To enable



- 22 -

searching the header from the logical beginning to the logical end (as opposed to physical beginning to end), the following aids are introduced. Referring now to Fig. 5, there is an HA (home address) pointer 70 saved in a known location in the frame header. This one byte pointer tells which entry in the header is the home address (HA) 70. This is used by the microcode to locate the logical beginning of the track in the cache. This HA pointer is written to the frame when the header is written. A wrap or end of frame entry is shown at 72. The wrap entry has a unique value in order to insure that it is not mistaken for an ordinary record entry. This entry informs the microcode to wrap to the beginning of the header to find the remainder of the record entries, i.e., this entry locates the end of the last record in the frame. A null entry is indicated at 74. This is a four byte entry embedded in the header that has the hexadecimal value "00000000." This entry, like the wrap entry, is guaranteed to be unique. The purpose of this entry simply is to signal that the end of the track has been reached. Thus, the header of Fig. 5, wherein records R4, R5 and R6 appear before the null entry 74 and home address 70, and records 1, 2 and 3 are placed between the home address 70 and the wrap entry 72, indicates that staging of this particular frame began with the fourth record on the track. Record 6 was the last record on the track, after which the null entry was written. Staging continued after staging of the home address pointer, and concluded with records 1, 2 and 3 after which the wrap entry, indicating the last record written to the physical cache frame was written.

Each of the entries for the records R1-R6 as shown comprises an ETP address, indicating the records starting location on the disk measured in bytes from the index mark and a word address "WA" indicating the starting location of the record within the cache memory. Correspondence is thus fully maintained between the locations of each individual record on the disk and in the cache memory. These entries, together with



- 23 -

the null, home address, and wrap entries comprise a header data field 70B. Also comprised in the header stored at the beginning of each frame in the cache is a header control field 70A. This comprises three entries, DID, (device identification), CYL (cylinder number of the staged track), HEAD (head number of the staged DASD track). These three entries clearly identify the location on the disk of the entire track stored in the frame. Also stored as part of the header control field 70A is the HAP, which is a pointer to the home address entry in the header data field 70B (the "known location" referred to above) so that if it is desired to locate the home address within the cache, it is not necessary to search through all the records stored previously thereto in the frame. Finally, comprised in the header control field 70A is a DL entry which indicates the length of the header data field. As noted above, one of the objects of the invention is to provide a variable length header for a data field stored in a solid-state array. The DL entry is necessary to indicate the ending of the header data field 70A.

As discussed above, after the entire record has been read from the disk drive into the cache, it is necessary to adjust the ETP values for that portion of the record stored after initiation of the read operation and before detection of the index mark, because the ETP values are not precisely known until the latter time. This may be done in a preferred embodiment of the invention by building the header in the control store conventionally comprised in the director 12. After the track has been entirely read into the cache, the header, complete and adjusted to reflect the exact ETP values for the records first stored, may be copied to the frame (in the cache memory 34) thus freeing the control store for future processing. This approach also provides the advantage that higher speed operations are possible than if the header entries were simultaneously being written to the cache.



- 24 -

It will be appreciated by those skilled in the art that this header format has a distinct advantage that only as many header entries need to be processed as actual records exist on disk in contradistinction to the method of, e.g., providing a header entry for each sector.

Fig. 6 shows schematically the image of the track, including the header, in the cache. The image begins with the frame header 60 (comprising the header control field 70A and the header data field 70B, of Fig. 5) which contains emulated track position values in bytes for beginning locations of the home address field and for record 0, for record 1, for record 2 and so on. The information staged from the DASD track follows the header.

When the host later desires to read a record stored in cache from disk, due to an error or the like, the emulated track position values stored in header data field 70B are used in re-establishing orientation. Fig. 4B shows registers which may be used to establish the orientation and illustrates the interplay between the control module 24 and the director 12 in this operation, while Figs. 7A, 7B and 7C comprise a detailed illustration of the track format and of certain timing diagrams which further explain how the reorientation operation works.

The discussion above in connection with Fig. 4A illustrated how the disk surface is divided into radially extending sectors 77 and circumferentially extending tracks 72. Fig. 7A shows a more detailed view of one of the tracks. The track is deemed to begin with an index mark 73 (found on the servo disk 26A) followed by a home address mark, a gap, followed by a field indicating the length of record 0, "R0 Count", another gap, "R0 Data", another gap, "R1 Count", another gap and so on. Fig. 7B shows a correspondingly scaled diagram of the sectors into which the track is divided. As will be noted, the sectors do not correspond with the beginnings of the records, nor are there an integral number of records per sector.



- 25 -

As discussed above in connection with Fig. 4A, the typical disk pack with which the invention is designed to be used comprises a number of disk surfaces for the storage of data and a dedicated disk surface 26A which contains permanently encoded sector information. The dedicated disk surface 26A can be used to provide an indication of a sector 77 in the vicinity of a record to be read, but cannot be relied upon to give an absolute indication of the place of beginning of a particular record. In prior art operations when an error was found in trying to locate the count field of a particular data record, it was essential to slip an entire revolution in an attempt to reorient on that same count field once again. If the same error occurred over and over again there was no way to obtain the data stored in the record, i.e., a permanent error was recorded. Recordation of the emulated track positions according to the method of the invention provides a way out of this difficulty. One can attempt to orient on the count field of the preceding record on the disk by using its ETP value stored in the header field, instead of looking for the count field of the record sought for. If an orientation can be established on the prior count field, then the succeeding count field will automatically be that of the record which is sought for. Accordingly, one source of permanent errors in disk drives is avoided by the generation of the ETP table according to the invention. It will be appreciated by those skilled in the art that if one desires to reorient operations from a cache to a disk, one must first locate the count field of the record which is, for example, being called for by a host, and that it is typically impossible to reorient on a particular count field in time to immediately supply the record to the host. Accordingly, if one simply output the emulated track position of the record which is to be located, one would have to wait an entire disk revolution before one could begin sending the data from the disk to the channel. Having generated a full table of ETP values, according to the invention, one can instead



- 26 -

reorient on the count field of the record preceding that of interest and can therefore begin reading data substantially immediately upon establishing orientation, thus speeding such reorient operations. Accordingly, when it is desired to reorient a read operation from the cache to the disk, the emulated track position of the record preceding the record of interest is used to locate its count field; thereafter the count field of the record sought is detected. This is explained in connection with Figs. 7A through 7C. Assuming that record 2 is to be located, the emulated track position of record 2 is first retrieved from the header data field 70B comprised in frame header field 60 (Fig. 5). The location is shown on the diagram as ETP_{R2} , and, as noted, occurs in sector 2. The sector is determined by dividing the ETP of R2 by the number of bytes per sector, and subtracting 1 from the result. Accordingly, the director looks for the beginning of sector 2 and is enabled at the beginning of sector 2, noted as "Director Waits" at 75 so that when the count field of record 2 is thereafter detected, the director is enabled to transfer the data from the disk to the host computer beginning at the data field of Record 2. Accordingly, the "director waits" time begins at the start of sector 2. A timing window 69 is calculated using ETP_{R2} , that is, at the point measured in bytes after the index mark, where it is anticipated that Record 2 will begin. Reading is begun, i.e., the "READ" command is issued to the drive, at the end of the director waiting period. At this time, the drive searches for the next count field. When coincidence is detected, reading of the data can begin.

The timing window is shown centered about the $ETPR2$ position in Fig. 7A. Since the beginning of the timing window must precede the desired count field, the timing window is located by calculating the number of whole sectors in the emulated track position and adding additional bytes to locate the center of the count field detection window. In other



- 27 -

words, the number of bytes per sector is divided into the emulated track position, so that the remainder, or residual value, is the number of bytes from the beginning of the next succeeding sector, sector 3 in Fig. 7A, about which the timing window is to be centered. After this calculation is performed, a number of bytes equal to half the width of the timing window is subtracted from the value thus derived and it is at this point, determined by the byte count, that the timing window opens, enabling the director to look for the beginning of the R2 Count field. Accordingly, the total number of ETP bytes, i.e., the number of bytes at which the record sought begins, measured from the index mark, is divided by the number of bytes per sector. For example, if the ETP is 325 bytes from the index mark and if a sector is 100 bytes long, then the ETP of the record sought for is 25 bytes past the beginning of the fourth sector (sector 3, counting from sector 0). If the timing window desired is 40 bytes wide, then it should extend from 5 to 45 bytes after the start of sector 3. The dedicated disk 26A is used to locate the beginning of sector 3; when 5 more bytes have passed, the timing window is "opened".

Hardware to implement the operation of the director during a reorientation operation as just described is depicted in Fig. 4B. Assume that it is desired to reorient on the R2 count field. As indicated at 61, the director selects from ETP table 60 the ETP of the previous record, record R1. (This is designated ETP of record N-1 in Fig. 4B.) The desired reorient count field is the one just prior to the field that requires processing by the channel, in this instance the R1 count field. Orienting on the prior count field enables processing by the channel to immediately resume without slipping a full track operation. The value of ETP_{R1} is set into the register 63. It is converted into an equivalent sector value and a residual value by dividing it by the number of bytes per sector in the disk. This conversion is performed by the divider 64. Assume as an example that the ETP_{R1} is 474. That is, 474 bytes



- 28 -

are in the gaps, the home address, the R0 count, and the R0 data fields prior to the R1 count field in Fig. 7A. Dividing 474 by 156 (the number of bytes in a sector) gives a quotient of 3 and a remainder (residual value) of 6. The quotient 3 identifies the target sector value as sector 2. (Since the numbering of sectors begins with 0, the third sector is sector 2.)

As indicated at 65, the director 12 sets the target sector value to 2, which is the equivalent sector value. The control module 24 of the disk drive compares this equivalent sector value with the sector signal provided from the dedicated surface 26A, as indicated at 66. As in response to a conventional SET SECTOR command, when the heads 27 of the disk pack reach sector 2, the control module produces an INTERRUPT SECTOR REACHED signal as shown in Fig. 7C. In response to the INTERRUPT SECTOR REACHED signal, the director 12 enables the reading of records from the disk after a further delay which is indicated at 67. This delay is calculated from the sector residual value. At the termination of this delay, indicated at the point 68 in Fig. 7A, the director 12 begins reading of records stored on disk to the central processing unit. This is accomplished by detecting the first count field which occurs within a timing window indicated at 69 in Fig. 7A. The timing window is established by enabling the address mark detection circuitry 71 in the drive for a duration of 20 track bytes. Since a count field can only occur every 186 track bytes, if a count field is encountered in this 20 track byte window, detection of the desired R2 count field is assured. A window is necessary for this read operation to ensure that the disk drive reliably detects the count field. When detection occurs, the SYNC IN signal is given.

In this manner, a program which processes data records may be reoriented from cache memory to the disk drive. This has been accomplished by the use of the table of ETP values. It should be noted that one advantage of the table of ETP



- 29 -

values is that it allows reorientation even though all of the bytes in the gaps have not been stored in the cache memory. By using the ETP values according to the present invention, reorientation is possible even though there is not a strict
5 correspondence between records on a disk track and the data copied to the cache memory 30.

It will be appreciated by those skilled in the art that the functions of the control module 24 are the same as performed in systems without caching capability, so that the
10 control module need not be modified to perform caching according to this invention and those described in the copending applications incorporated by reference above. Indeed the lack of modification to the control module 24 is a desired object of these inventions.

It will be appreciated that the reorientation of data flow from cache to disk is shown in Figs. 4B and 7. It is essentially the inverse of the process of copying a track to
15 cache as shown in Fig. 4A. In both cases, a sector address is first looked for, and after the proper sector is located, then the header address is compared with a target header stored in a register so as to determine when the correct header has been
20 reached. This is done in the case of reorientation to allow reading or writing data from the disk to commence at the correct point, and in the case of copying data from the disk to the cache (Fig. 4A) to give the indication that a full track
25 has been thus copied. The reorientation process described in connection with Figs. 4B and 7 is slightly complicated by the fact that the sector preceding that containing the record at which reading from the disk is desired to begin must be looked
30 for because the record's starting location or disk cannot be assumed to be coincident with the beginning of the sector within which they are located.

As discussed in copending application Serial No. 325,287 filed November 27, 1981, of Blickenstaff et al,
35 entitled "Data Path Reconfiguration for Error Control",



- 30 -

incorporated herein by reference, in the event of an error indication generated by the cache manager, it is important that "orientation" or data path establishment be redirected from the cache to the disk device so that the cache can be "cut out" of the data path, thus returning the path to that which would have existed if the cache had never been added to the long-term memory storage system. This must be done as discussed above in such a way that the addition of the cache requires no modifications to the host, i.e., that it be plug compatible and software transparent thereto. Accordingly, when the system of the invention detects an error in the cache system it is inappropriate to terminate the host's request with the indication that the operation cannot be completed. In fact, the operation can be completed on the disk device to which the host's request was ostensibly addressed, thus fulfilling the goal of software transparency mentioned above. The problem with switching the operation to the disk drive is one of insuring that the operation begins at the correct starting address on disk. Superficially, it would appear that it would be sufficient to compare the record identifier of the disk record supplied by the host against the identifier from the operation in the cache. However, there is no guarantee that all the record identifiers on the track are unique. A data integrity problem could thus be presented. This is overcome according to the present invention by using both the record identifier, which is an integral part of each record, and the ETP value associated with the starting point of the record on a track. Both variables are available at the time the cache is being used to satisfy the host's request.

As an example, attached hereto as Appendix A and incorporated by reference is a microcode-language document entitled "STAGE REORIENT" which details the way in which reorientation as described above is accomplished. This document, together with the above description of the invention and the copending applications referred to and incorporated by



- 31 -

reference above, will enable those skilled in the art to practice it without difficulty.

It will be appreciated that there has been described a method of control of a cache buffer memory which satisfies the needs of the art and objects of the invention listed above. By generating emulated track position information as to each record as data is read from disk into cache, one has available at all times an accurate correlation between disk and present cache address if for some reason it is desired to switch channel orientation back to the disk from the cache. Furthermore, correct maintenance of the ETP register allows one to know accurately when the entire disk track has been read. This in turn permits one to write data from disk to cache without waiting for the index mark to be juxtaposed to the read/write head. Finally, provision of a frame header generated when the data is written to the cache permits the header to be of a length proportional to the number of records in the cache, while providing a direct correspondence between the location of records in the cache and their locations on disk.

Those skilled in the art will likewise recognize that there are numerous improvements and modifications that can be made to the invention without departure from its essential spirit and scope which is therefore not to be limited by the above exemplary disclosure of the preferred embodiment, but only by the following claims.



EC 90039

ADDR	INST
000000	000000
000001	000001
000002	000002
000003	000003
000004	000004
000005	000005
000006	000006
000007	000007
000008	000008
000009	000009
000010	000010
000011	000011
000012	000012
000013	000013
000014	000014
000015	000015
000016	000016
000017	000017
000018	000018
000019	000019
000020	000020
000021	000021
000022	000022
000023	000023
000024	000024
000025	000025
000026	000026
000027	000027
000028	000028
000029	000029
000030	000030
000031	000031
000032	000032
000033	000033
000034	000034
000035	000035
000036	000036
000037	000037
000038	000038
000039	000039
000040	000040
000041	000041
000042	000042
000043	000043
000044	000044
000045	000045
000046	000046
000047	000047
000048	000048
000049	000049
000050	000050
000051	000051
000052	000052
000053	000053
000054	000054
000055	000055
000056	000056
000057	000057
000058	000058
000059	000059
000060	000060
000061	000061
000062	000062
000063	000063
000064	000064
000065	000065
000066	000066
000067	000067
000068	000068
000069	000069
000070	000070
000071	000071
000072	000072
000073	000073
000074	000074
000075	000075
000076	000076
000077	000077
000078	000078
000079	000079
000080	000080
000081	000081
000082	000082
000083	000083
000084	000084
000085	000085
000086	000086
000087	000087
000088	000088
000089	000089
000090	000090
000091	000091
000092	000092
000093	000093
000094	000094
000095	000095
000096	000096
000097	000097
000098	000098
000099	000099
000100	000100
000101	000101
000102	000102
000103	000103
000104	000104
000105	000105
000106	000106
000107	000107
000108	000108
000109	000109
000110	000110
000111	000111
000112	000112
000113	000113
000114	000114
000115	000115
000116	000116
000117	000117
000118	000118
000119	000119
000120	000120
000121	000121
000122	000122
000123	000123
000124	000124
000125	000125
000126	000126
000127	000127
000128	000128
000129	000129
000130	000130

EC 90039

▼ DTIC-164 ▼

APPENDIX A

```

** MICROCODE ASSEMBLER 6.0  08/28/82  18:21:14  **  PAGE      JK100.001
PROGRAM:  JK100

```

Stage Reorient

STATEMENT

START OF SPECIFICATIONS

MODULE NAME: JX100

MODULE TITLE: Stage Reorient

FUNCTIONAL DESCRIPTION

This module regains orientation on the DASD track. The registers in the Stage Orientation Group (SOG) and the Stage Control Group (SCG) are initialized by this module.

No parameters are explicitly passed to or returned from this module.

OPERATIONAL DESCRIPTION

The Current ETP is initialized to zero, and the Header Entry Pointer is set to point to the beginning of the first entry in the frame header buffer area.

The current sector value is read from the DASD and converted into the Target ETP value and saved in the Stage Orientation Group (SOG). To find the next count field, a Read Q3 AM Search is executed on DASD via the following steps:

- 1) The Set Read/Write Mode module is called.
- 2) Issue a Read Q3 AM Search tag.
- 3) Call the Wait for DASD Tag Valid module.
- 4) Set up the DASD transfer controls.
- 5) Wait for DASD end of transfer.
- 6) Compare the physical address bytes from the Group C registers with the physical address bytes in the Data Buffer count field area. If they are not the same, take an error exit.
- 7) Save the 'record number' byte from the count field as the Target Record Number in the Stage Control Group (SCG). Initialize the defect bits in the Orientation Control Byte (SCG) using bits from the count field flag byte. Save the 'SDI' bytes from the count field as the Current Skip Defect bytes (SOG). The Current field Length bytes (SOG) are initialized to zero.
- 8) Call the Wait for DASD Normal End module.

If no count field is encountered by the time index comes up in the DASD, then the 'reorient point at index' bit is set and the Target ETP is set to the maximum value.

SPECIAL CONSIDERATIONS

None.

** SYC SECRET **
 PN 500703B-01
 PAGE JX100.0001

SUBSTITUTE SHEET



JX100.002

PAGE

PN 5007038-01

.. MICROCODE ASSEMBLER 6.8 06/25/82 18:21:14 ..

PROGRAM: JX100

Stage Reorient

EC 90039

ADDR INST

INTERNAL ITEMS	NAME	DESCRIPTION/PARAMETERS	STMT #
ENTRY POINT(S):	JX100	-----	53
NORMAL EXIT(S):	caller	-----	54
ERROR EXIT(S):	JX995	-----	55
		-----	56
		-----	57
		-----	58
		-----	59
		-----	60
		-----	61
		-----	62
		-----	63
		-----	64
		-----	65
		-----	66
		-----	67
		-----	68
		-----	69
		-----	70
		-----	71
		-----	72
		-----	73
		-----	74
		-----	75
		-----	76
		-----	77
		-----	78
		-----	79
		-----	80
		-----	81
		-----	82
		-----	83
		-----	84
		-----	85
		-----	86
		-----	87
		-----	88
		-----	89
		-----	90
		-----	91
		-----	92
		-----	93
		-----	94
		-----	95
		-----	96
		-----	97
		-----	98

Stage Error Exit
Time-out waiting for Address Mark
Error while waiting for EOT
Physical address mismatch on stage
DASD index reset time-out
Wait for DASD Tag Valid
Wait for DASD Normal End
Get Current DASD Sector
Set Read/Write Mode
Multiply
Set Secondary IRG
Stage Control Group (SCG)
Stage Orientation Group (SOG)
Data Buffer count Field (DBC)
Group C registers
current group not affected
Internal Register Group
Device Tag Date
Device Tag Out
Device Bus-out
Device Transfer Counter High
Device Transfer Counter Low
Device Transfer Control
Buffer Input Pointer
Buffer Output Pointer
Maintenance Control/Sense
Transfer Complete Status
Device Tag In
Buffer Address Pointer
Buffer Increment Register
EXIT IRG: Pri - restored
Sec - SCG
ENTRY IRG: Pri - <7
Sec - SCG
CHANGE ACTIVITY
02/10/82 BET Added error exit parameters
03/17/82 BET Added chained stage enhancement
06/09/82 BET Implemented fix for reorient on record 1
07/06/82 BET Lengthened AM time-out timer
..... END OF SPECIFICATIONS

JX' 702

PAGE

PN 5007038-01

.. STC 5 T ..

EC 90039

SUBSTITUTE SHEET



PAGE JX100.004

PN 5007038-01

.. MICROCODE ASSEMBLER 6.8 08/25/82 18:21:14 ..
 PROGRAM: JX100

Stage Reorient

EC 90039

ADDR	INST	STMT #
3433 DBA2	S2 = \$X86500SC	146
3434 BAB0	S3 = \$X86500SC	147
3435 JAE	DTG = X0E	148
3436 JAE	DTG = X0E	149
3437 JAE	DTG = DTG / X02	150
3438 JAE	DTG = DTG / X10	151
3439 JAE	DTG = DTG . XEF	152
343A 67B5	BAL JX900	153
	.. Set the DASD data transfer counters (DCH, DCL registers) ..	154
	.. Set the Buffer Input and Buffer Output pointers (BIP & BOP registers) ..	155
	.. Set the Input priority mode (DKC register) on for the Data Buffer ..	156
	.. Set the DASD read/write/clock control bits (DTG register) for a read ..	157
	.. Set the remaining CTL-1 transfer control bits (DTG, MCS, DKC regs) ..	158
	.. Set the DASD transfer counters ..	159
	DCH = X00	160
	DCL = X10	161
	DKC = DKC / X02	162
	PO = XEE	163
	BIP = PO	164
	BOP = PO	165
	DKC = DKC / X02	166
	DTG = DTG . XFO	167
	DTG = DTG / X0A	168
	DTG = DTG / X01	169
	MCS = MCS . XCF	170
	MCS = MCS / X30	171
	DKC = DKC / X01	172
	.. Set EDT-Timer (local) for 20 milliseconds ..	173
	.. DO-WHILE waiting for DASD end of transfer (XCS register) ..	174
	.. Decrement EDT-Timer ..	175
	.. IF EDT-Timer is zero ..	176
	.. THEN TRANSFER to Stage Error Exit ..	177
	.. ELSE NULL ..	178
	.. Set EDT timer for 20 milliseconds ..	179
	.. (that's 9,081 loops, folks!) ..	180
	.. Test DASD EDT bit ..	181
	.. Branch if EDT is up ..	182
	.. Decrement the timer ..	183
	.. Branch if timer is not expired ..	184
	.. Branch to error exit routine ..	185
	.. Time-out waiting for AM ..	186
	.. IF Index (DTI register) is encountered ..	187
	.. Set EDT timer for 20 milliseconds ..	188
	.. (that's 9,081 loops, folks!) ..	189
	.. Test DASD EDT bit ..	190
	.. Branch if EDT is up ..	191
	.. Decrement the timer ..	192
	.. Branch if timer is not expired ..	193
	.. Branch to error exit routine ..	194
	.. Time-out waiting for AM ..	195
	.. IF Index (DTI register) is encountered ..	196

PAGE JX100.004

PN 5007038-01

.. STC 5 T ..

EC 90039

SUBSTITUTE SHEET



PAGE JX100.005

PN 5007038-01

** MICROCODE ASSEMBLER 6.0 08/25/82 16:21:14 **

EC 90039

PROGRAM: JX100

Stage Reorient

ADDR	INST	STATEMENT	STMT #
3454	38D2	THEN-DO for index encountered	197000
3455	416A	Set 'reorient point at index' flag (SCG)	198000
3456	9281	Set Target-ETP to the out of range value (XFFFF)	199000
3457	F8AF	Set Index-ETP (local) for 15 microseconds	200000
3458	F8BF	CHK_INDX D = DTI TM X02	201000
3459	8A2A	BZ	202000
		*Test index latch	203000
		*Branch if it is down	204000
		*Set 'reorient point at index' flag	205000
		*Set Target ETP out of range	206000
		P2 = XFF	207000
		*Set index reset loop timer	208000
		DO-WHILE index latch is up	209000
		Decrement Index-ETP	210000
		IF Index-ETP equals zero	211000
		THEN TRANSFER to Stage Error Exit	212000
		PARAM LIST: DASD index reset time-out	213000
		ELSE Toggle 'index latch reset' bit	214000
		END-WHILE index latch is up	215000
		Send response to the DASD	216000
		Reset run auto (DXC register) and run device (DTG reg) bits	217000
		RETURN for reorient point at index	218000
		END-THEN for index encountered	219000
		ELSE NULL	220000
		CHK_DROP D = DTI TM X02	221000
		BZ	222000
		SEND_RSP	223000
		P2 = P2 + XFF	224000
		BN2	225000
		RST_INDX	226000
		BAL	227000
		DC X8E45	228000
		DC X9245	229000
		DC X9245	230000
		RST_INDX DTG = DTG / X10	231000
		DTG = DTG / XFF	232000
		CHK_DROP	233000
		SEND_RSP DTG = DTG / X20	234000
		DTG = DTG / XDF	235000
		DTG = DTG / XFE	236000
		DXC = DXC / XFE	237000
		EXIT	238000
		IF check-2 (XCS register) is on	239000
		check and (DTI register) is on and index is off	240000
		THEN TRANSFER to Stage Error Exit	241000
		PARAM LIST: Error while waiting for EDT	242000
		ELSE NULL	243000
		END-WHILE waiting for EDT	244000
			245000
			246000
			247000

PAGE JX100.005

PN 5007038-01

** STC SECRET **

EC 90039

SUBSTITUTE SHEET



- 37 -

EC 90039

PN 5007038-01

** MICROCODE ASSEMBLER 6.0 08/25/82 18:21:14 **
PROGRAM: JX100

EC 1

PAGE

JX1

JX1

JX1

JX1

JX1

JX1

JX1

JX1

JX1

JX1

JX1

Stage Reorient

ADDR	INST	STATEMENT	STAT #
346A	1005	CHK_CEC2 JH XCS O EOT_ERR	248000
346B	20D2	D = DTI TM X20	249000
346C	414B	BZ CHK_EOT	250000
346D	38D2	D = DTI TM X02	251000
346E	434B	BNZ CHK_EOT	252000
346F	64D0	BAL JX995	253000
3470	8E43	DC XBE43	254000
3471	38D1	DA NX005	255000
3472	9243	DC X9243	256000

		* Reset run auto (DXC register) and run device (DTG register) bits	257000
		* If the physical address bytes from the Data Buffer count field area	258000
		* don't match the physical address bytes in the Group C registers	259000
		* THEN TRANSFER to Stage Error Exit	260000
		* PARM LIST: Physical address mismatch on a stage	261000
		* ELSE NULL	262000

3473	30CE	EOT_UP DTG = DTG XFE	263000
3474	3038	DXC = DXC XFE	264000
		XSETSI RG XOC	265000
3475	31F0	IRG = IRG XFO	266000
3476	33FC	IRG = IRG / XOC	267000
3477	2A2F	BAP = XFO	268000
3478	3224	BAP = BAP / X04	269000
3479	2553	P3 = BFI	270000
347A	7C83	D = P3 CE S0	271000
347B	4882	BNEQ PA_MISC	272000
347C	2C53	P3 = BFI	273000
347D	7C93	D = P3 CE S1	274000
347E	4882	BNEQ PA_MISC	275000
347F	2C53	P3 = BFI	276000
3480	7C43	D = P3 CE S2	277000
3481	4988	BEQ PA_OK	278000
		XSETSI RG STOCG	279000
3482	31F0	IRG = IRG XFO	280000
3483	33F9	IRG = IRG / X09	281000
3484	64D0	BAL JX995	282000
3485	8E44	DC XBE44	283000
3486	38D1	DA NX005	284000
3487	9244	DC X9244	285000
		XSETSI RG STOCG	286000
3488	31F0	IRG = IRG XFO	287000
3489	33F9	IRG = IRG / X09	288000

		* Set defect bits in the Orientation Control Byte from the flag byte	289000
		* Set Target-Record-Number (SCG) - record number from DB count area	290000
		* IF key length byte in the Data Buffer count area is nonzero	291000
		* THEN Set 'key field next' flag (SCG) on	292000
		* ELSE Set 'data field next' flag (SCG) on	
		* Set Current-Skip-Defect-Bytes (SDG) = SDI bytes from DB count area	

** STC 5 IV **

EC 1

PAGE

JX1

JX1

JX1

JX1

JX1

JX1

JX1

JX1

JX1

JX1

JX1

SUBSTITUTE SHEET



- 38 -

EC 90039

ADDR INST

348A 2053
 348B F030
 348C 818F
 348D 7238
 348E 3624
 348F 2059
 3490 2053
 3491 4194
 3492 8284
 3493 0495
 3494 8282
 3495 31F0
 3496 33FD
 3497 222F
 3498 3222
 3499 2058
 349A 2059
 349B BAA0
 349C BAB0
 349D 31F0
 349E 33F9
 349F 6871
 34A0 27FF
 34A1 4C54
 34A2

.. MICROCODE ASSEMBLER 6.8 08/25/82 18:21:14 ..
 PROGRAM: JX100

PAGE JX100.007

PN 5007038-01

Stage Reorient

STATEMENT

STMT #

* Set Field-Length-Bytes (SOG) to zero

 P3 = BFI
 P3 = P3 . XEO
 SO = SO . XIF
 SO = SO / P3
 BAP = BAP + X04
 S1 = BFI
 P3 = BFI
 B2
 SET_DFN
 SO = SO / X04
 B
 SET_CSDM
 SET_CSDM XSETIRG ST000
 IRG = IRG . XFO
 SET_CSDM IRG = IRG / X0D
 BAP = BAP / X02
 SO = BFI
 S1 = BFI
 S2 = X00
 S3 = X00
 XSETIRG ST000
 IRG = IRG . XFO
 IRG = IRG / X09

 * CALL Wait for DASD Normal End module
 * PASS: nothing
 * RECEIVE: DASD bus-in value
 * RETURN for reorient point at a count field

 EXIT
 IRG = IRG + XFO
 BALR P4P5,P4P5
 END

2930000 293
 2940000 294
 2950000 295
 2960000 296
 2970000 297
 2980000 298
 2990000 299
 3000000 300
 3010000 301
 3020000 302
 3030000 303
 3040000 304
 3050000 305
 3060000 306
 3070000 307
 3080000 308
 3090000 309
 3100000 310
 3110000 311
 3120000 312
 3130000 313
 3140000 314
 3150000 315
 3160000 316
 3170000 317
 3180000 318
 3190000 319
 3200000 320
 3210000 321
 3220000 322
 3230000 323
 3240000 324

EC 90039

.. STC SECRET ..

PN 5007038-01

PAGE JX100.007

SUBSTITUTE SHEET



- 39 -

EC 90039		** MICROCODE ASSEMBLER 0.8 09/25/82 18:21:14 **		PN 5007038-01		PAGE		JX100.008	
PROGRAM: JX100		Stage Reorient		PN 5007038-01		PAGE		JX	
** LABEL TABLE WITH REFERENCES **		REFERENCE ADDRESSES		PN 5007038-01		PAGE		JX	
LABEL	ADDR	EQUATE VALUE							
BAP		E2	3477 3478 3479 3480 3481 3482 3483 3484						
BFI		E5	3479 347C 347F 348A 348F 3490 3498 349A						
BIP		E0	3440						
BOP		E1	3441						
CHK_CEC2	348A		3458						
CHK_DROP	348A		3464						
CHK_EDT	344B		346C 346E						
CHK_INDX	3454		344F						
DBD		E9	3436 3437 3437						
DCH		E10	3438						
DCL		E11	343C 343D 343D						
DTG		E12	3429 3429 3438 3438 3438 3439 3443 3443 3444 3444 3445 3445 3462 3462 3463 3463						
DTI		E13	3429 3429 3438 3438 3438 3439 3443 3443 3444 3444 3445 3445 3462 3462 3463 3463						
DTO		E14	3429 3429 3438 3438 3438 3439 3443 3443 3444 3444 3445 3445 3462 3462 3463 3463						
DXC		E3	3429 3429 3438 3438 3438 3439 3443 3443 3444 3444 3445 3445 3462 3462 3463 3463						
EDT_ERR	346F		3429 3429 3438 3438 3438 3439 3443 3443 3444 3444 3445 3445 3462 3462 3463 3463						
EDT_UP	3473		3429 3429 3438 3438 3438 3439 3443 3443 3444 3444 3445 3445 3462 3462 3463 3463						
EXIT	34A0		3429 3429 3438 3438 3438 3439 3443 3443 3444 3444 3445 3445 3462 3462 3463 3463						
FH8DF1			3429 3429 3438 3438 3438 3439 3443 3443 3444 3444 3445 3445 3462 3462 3463 3463						
IRG			3429 3429 3438 3438 3438 3439 3443 3443 3444 3444 3445 3445 3462 3462 3463 3463						
JX100		X2425	3427 3428						
JX900		E31	3420 3420 3421 3421 3422 3422 3425 3425 3425 3425 3425 3425 3425 3425 3425 3425						
JX945		X3420	3483 3483 3488 3488 3488 3489 3489 3489 3489 3489 3489 3489 3489 3489 3489						
JX955		X3785	3420						
JX985		X3871	343A						
JX985		X39C8	349F						
JX995		X37DB	342A						
KX130		X3ADO	3430						
MCS		X3F88	3450 345E 346F 3484						
NOTE01		E26	342C						
NKO05		X3BD1	3446 3446 3447 3447						
PA_MISC	3435		3432						
PA_OK	3482		3452 3460 3471 3486						
PO	3488		3478 347E						
P1		P0	3481						
P2		P1	343F 3440 3441 3449 344E 344E						
P2P3		P2	342B 344A 344D 344D						
P3		P3	3459 345C 345C						
P4P5		P3	342D						
RST_INDX	3462		3479 347A 347C 347D 347F 3480 348A 348B 348B 348D 3490						
SEND_RSP	3465		34A1 34A1						
SET_CSD8	3495		345D						
SET_OFN	3494		3458						
STGCD		X09	3493						
STGDD		X0D	3491						
SI		SO	3425 3482 3488 349D						
		S1	3421 3495						
			3456 3456 347A 348C 348C 348D 348D 3492 3492 3492 3494 3494 3499						
			347D 348F 349A						
EC 1		** SYC : T **		PN 5007038-01		PAGE		JX	

SUBSTITUTE SHEET



JX100.009

PAGE

PN 5007038-01

18:21:14 **

08/25/82

MICROCODE ASSEMBLER 6.8

PROGRAM: JX100

EC 90039

EC 90039

Stage Reorient

** LABEL TABLE WITH REFERENCES **

REFERENCE ADDRESSES
342F 342F 3431 3433 3457 3480 3498
342D 342E 342E 3434 3458 349C
3423 3427
3424 3428
3448 346A
342E 342F 3433 3434

EQUATE VALUE	ADDR	LABEL
S2		S2
S253		S253
S3		S3
S4		S4
S5		S5
XCS		XCS
X86500SC		X86500SC
X820C		X820C

JX100.009

PAGE

PN 5007038-01

18:21:14 **

08/25/82

MICROCODE ASSEMBLER 6.8

PROGRAM: JX100

EC 90039

EC 90039

** STC SECRET **

SUBSTITUTE SHEET



We claim:

1. A cache memory subsystem for use with magnetic disk data storage media, said cache memory subsystem comprising:

a solid-state memory array; and

5 controller means, said controller being adapted to maintain information whereby correspondence between data locations on said disk storage media and storage addresses of data records stored in said solid-state array can be maintained, said correspondence being established by comparison
10 of an emulated track position to the starting address of said record in said cache memory, said emulated track position corresponding to the point on a given disk track at which a given data record stored in said cache memory means begins.

2. The subsystem of claim 1 wherein said emulated track position value is stored in a first register at the beginning of writing of data from a disk track into said cache, a second register being initialized to zero when an index mark
5 is detected on said disk track, said second register being thereafter incremented upon reading of each byte from said disk track to said cache, whereby the contents of said second register may be compared to said emulated track position, the values being the same when an entire track has been read.



3. In a method of control of a cache memory subsystem adapted to store information received from magnetic disk memory media in quantities equal to the capacity of one disk track, wherein said data is read sequentially starting at any position on a given disk track, the improvement which comprises:

in a first register, storing an emulated track position corresponding to the relative position on said disk at which said data begins to be read into said cache;

setting a second register to zero upon detection of an index mark on said disk during reading of said data from said disk into said cache;

incrementing said second register upon reading of successive bytes of data; and

comparing the contents of said second register with the emulated track position stored in said first register, and when said contents equal said emulated track position, providing an indication that an entire disk track has been read into said cache.

4. The method of claim 3 further comprising the step of writing header information from the first record read from disk into said cache in a given operation into a third register, comparing the contents of said third register with header information stored on said disk as part of records written to said cache, and terminating said read operation when correspondence is found.

5. In a method of staging data stored on disk memory media in logical tracks made up of plural records into solid-state memory means, the improvement which comprises writing identifying header information into a header directory in said cache during said staging, wherein the number of entries in said header directory is proportional to the number of said records comprised in a given disk track.



6. The method of claim 5 wherein said header information additionally comprises bytes indicative of the point at which an index mark on said disk is detected during said staging of data to said cache.

7. A cache memory subsystem for a data processing system in which a central processing unit processes data records stored on magnetic disk units, comprising:

a solid-state cache memory;

5 means for storing a table of emulated track positions each representing the equivalent rotational position at which data records, stored in said cache memory, are stored on a track of a magnetic disk; and

10 means for reorienting the operation of a program from records stored in said cache memory to corresponding records stored on a disk unit in response to the emulated track position of the record with respect to which the reorientation occurs, whereby said reorientation is performed within one revolution of the magnetic disk on which said corresponding
15 records are stored.

8. The cache memory subsystem recited in claim 7 wherein said emulated track positions each represent the number of bytes after the index mark on the disk track at which storage of the corresponding record commences.

9. The cache memory subsystem recited in claim 7 wherein said means for storing includes means for initially storing in said table the sector number of the first record stored in said cache memory, said number being thereafter
5 incremented as each byte is read from a track on a magnetic disk to produce a series of emulated track position values, each representing the number of bytes from the index mark at which the corresponding record stored in said cache memory begins on the disk.



10. The cache memory subsystem recited in claim 7 wherein said means for reorienting includes:

means for selecting from said table the emulated track position of the record just prior to the record stored on disk which is to be processed by a program.

11. The cache memory subsystem recited in claim 7 wherein said means for reorienting includes:

means for converting said emulated track position into an equivalent sector value.

12. The cache memory subsystem recited in claim 11 further comprising:

means for dividing the emulated track position by the number of bytes per sector in said disk to generate an equivalent sector value.

13. The cache memory subsystem recited in claim 11 wherein each magnetic disk unit has means for producing a sector signal representing the sector of the magnetic disk which is under the magnetic heads of the disk unit;

means for comparing the sector signal with said equivalent sector value; and

means for enabling the reading of records stored on said disk track to said central processing unit in response to said means for comparing.

14. The cache memory subsystem recited in claim 13 wherein said means for converting includes means for producing a sector residual value representing the number of bytes after said sector value at which the record, upon which reorientation occurs, is stored.



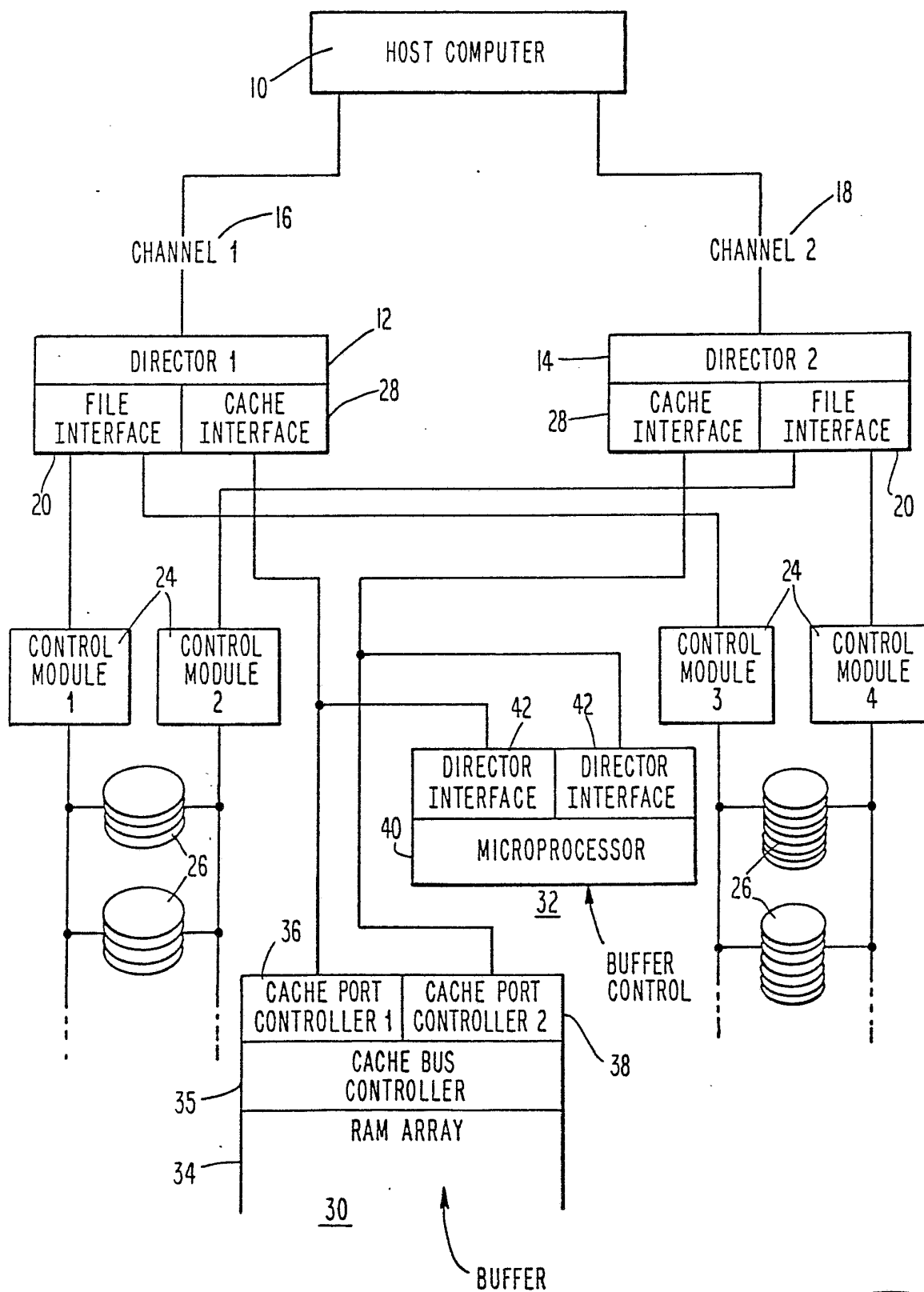
15. The cache memory subsystem recited in claim 14 further comprising:

means for delaying said means for enabling for a time determined by said sector residual value.

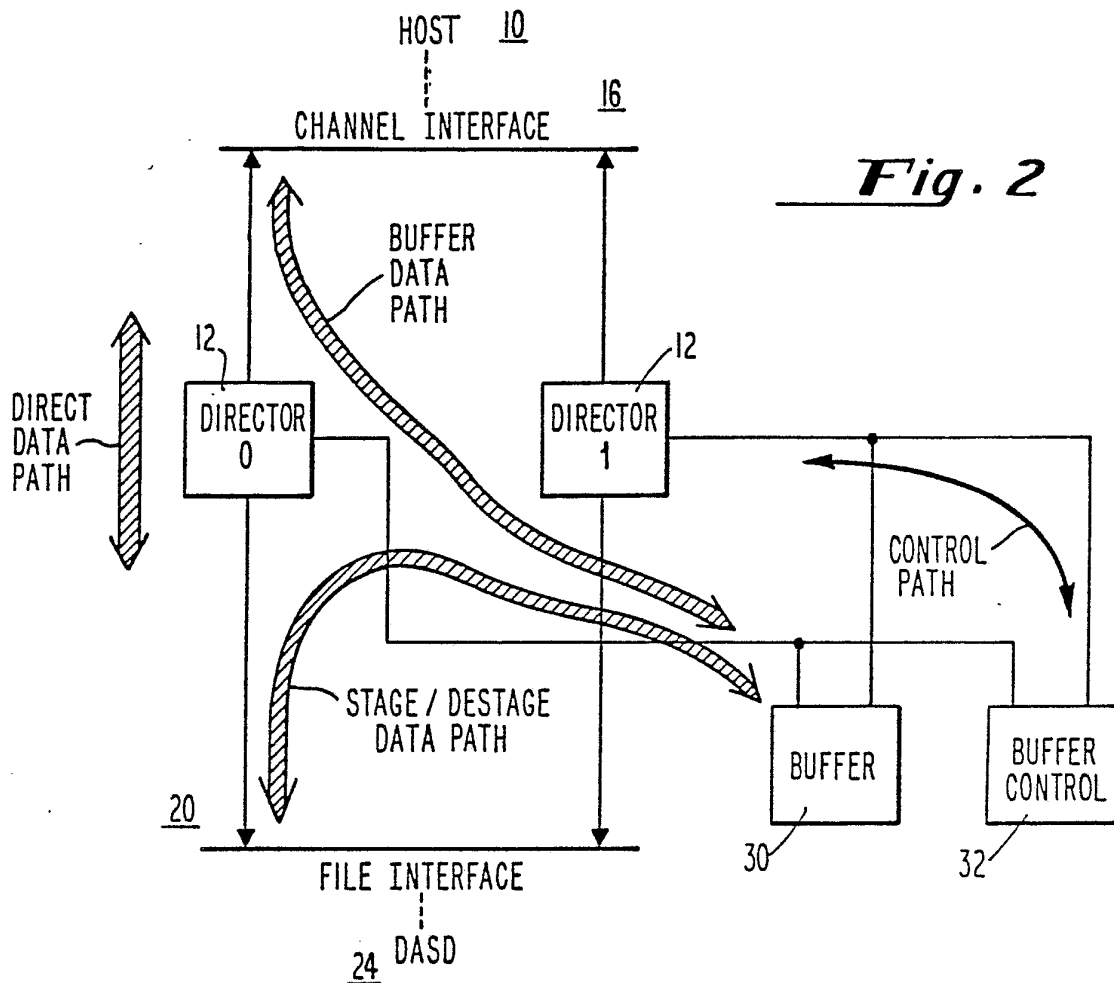
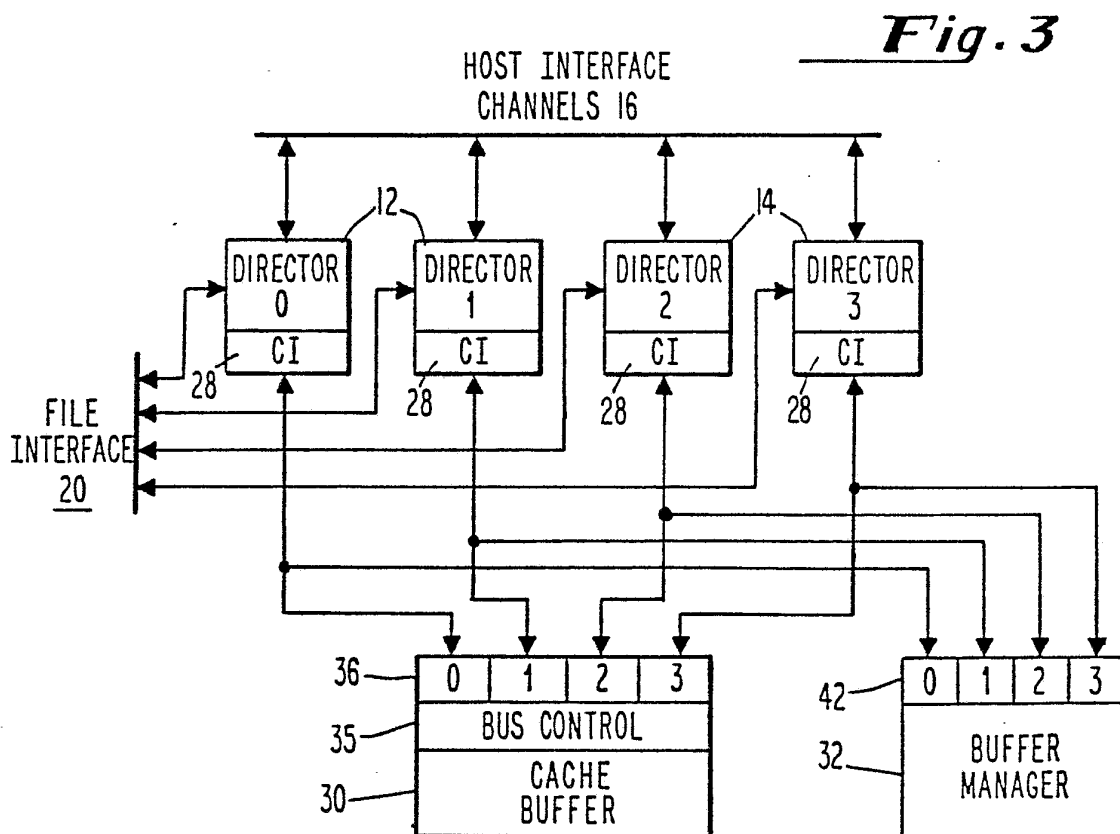
16. The cache memory subsystem recited in claim 13 wherein said means for enabling is operable for a given time window; and

5 means for detecting the count field of the record desired for reorientation is enabled during said time window.



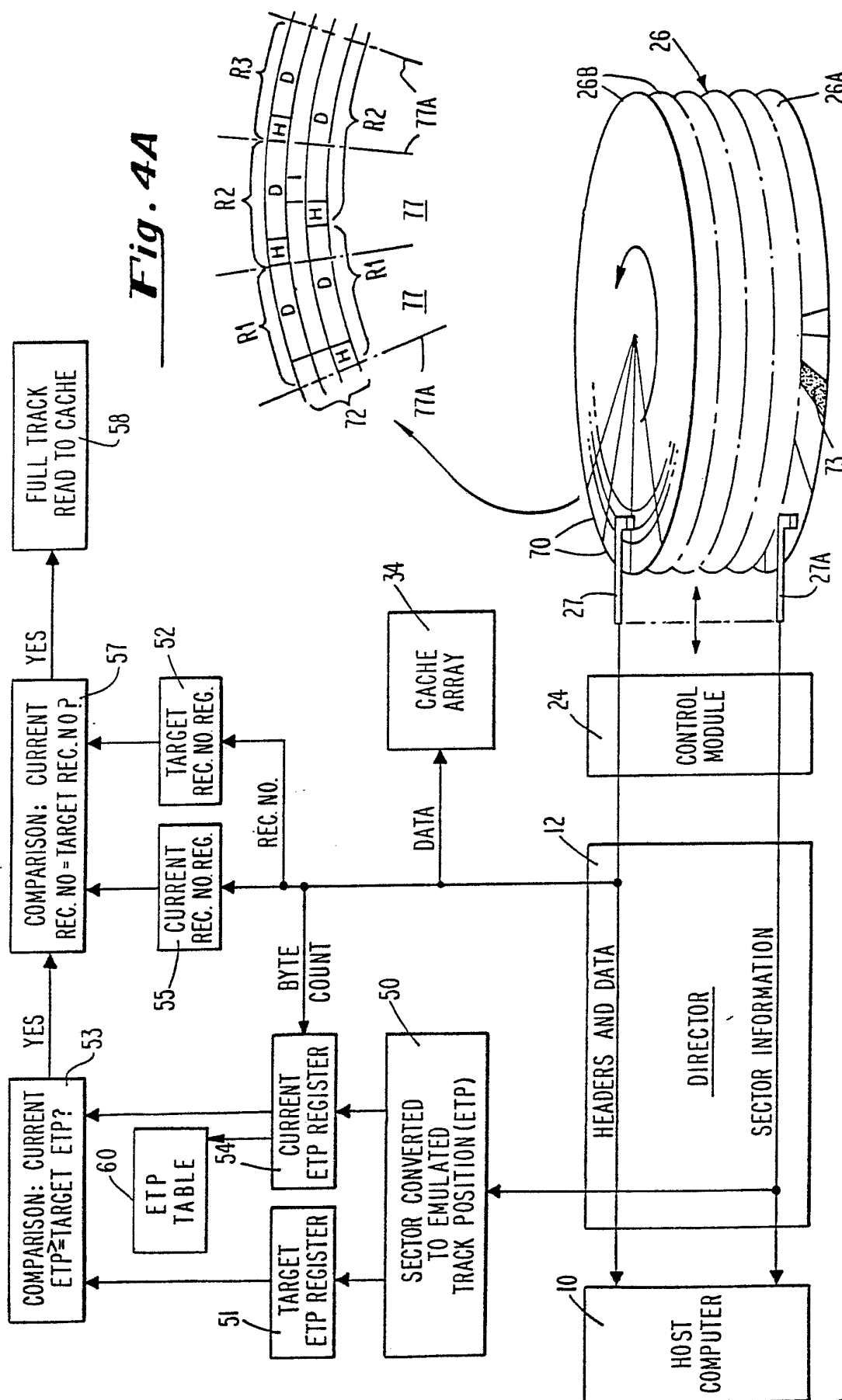
Fig. 1

2 / 5



SUBSTITUTE SHEET





4/5

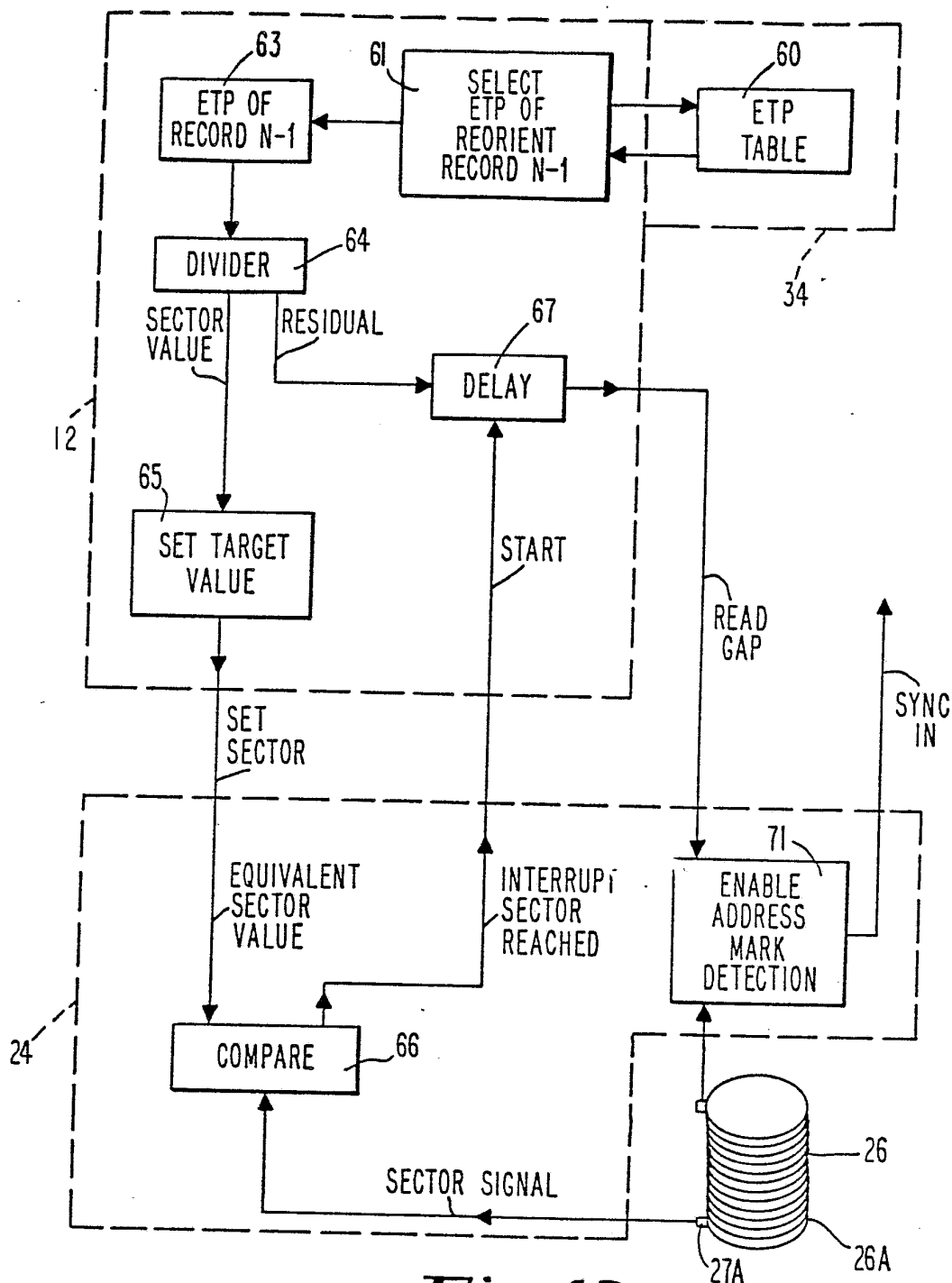


Fig. 4B

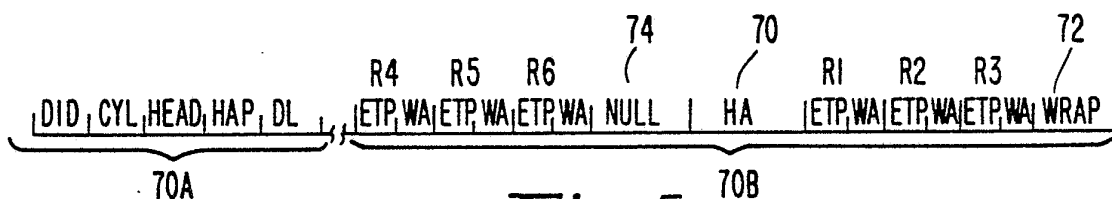


Fig. 5

SUBSTITUTE SHEET



5/5

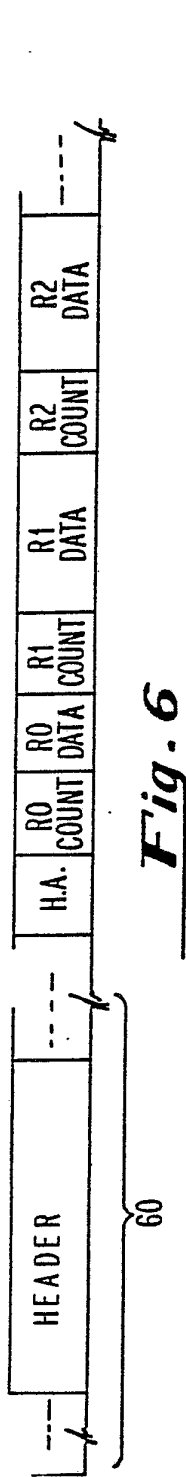


Fig. 6

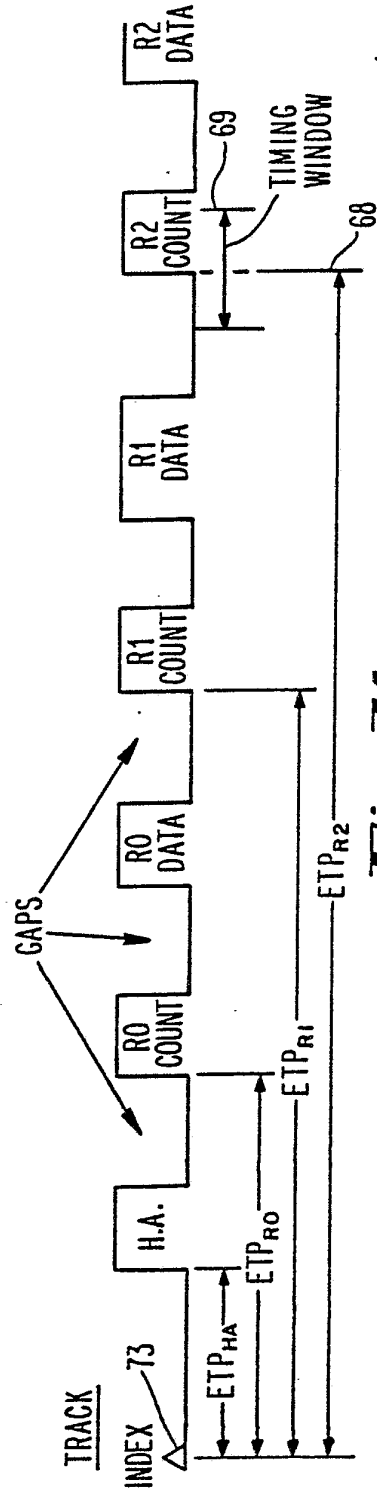


Fig. 7A

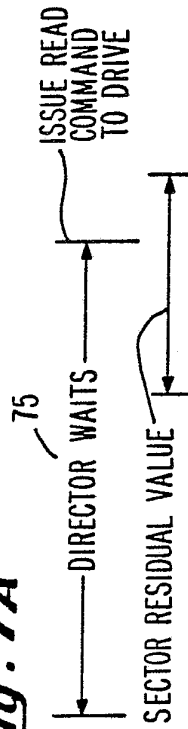


Fig. 7B

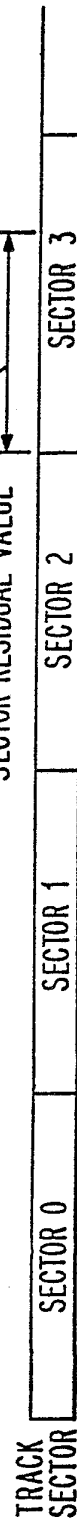


Fig. 7C

INTERRUPT SECTOR REACHED

INTERNATIONAL SEARCH REPORT

International Application No. PCT/US83/01770

I. CLASSIFICATION OF SUBJECT MATTER (If several classification symbols apply, indicate all) ³		
According to International Patent Classification (IPC) or to both National Classification and IPC INT. CL. G06F 13/04, 13/06, 3/06, U.S. CL. 364/200, 900; 360/50; 371/10, 38		
II. FIELDS SEARCHED		
Minimum Documentation Searched ⁴		
Classification System	Classification Symbols	
364 360 371	200,900 50 10, 38	
Documentation Searched other than Minimum Documentation to the Extent that such Documents are Included in the Fields Searched ⁵		
III. DOCUMENTS CONSIDERED TO BE RELEVANT ¹⁴		
Category ⁶	Citation of Document, ¹⁶ with indication, where appropriate, of the relevant passages ¹⁷	Relevant to Claim No. ¹⁸
Y, P	US, A, 4,394,733, (Swenson) 19 July 1983	1, 5-13
Y	US, A, 4,241,420, (Fish et al) 23 December 1980	1, 5-13
Y	US, A, 4,075,686, (Calle et al) 21 February 1978	7-13
A	US, A, 4,007,448, (Sergeant et al) 08 February 1977	1-16
A	US, A, 3,566,358, (Hasbrouck) 23 February 1971	1-16
<div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <p>[*] Special categories of cited documents: ¹⁵</p> <p>"A" document defining the general state of the art which is not considered to be of particular relevance</p> <p>"E" earlier document but published on or after the international filing date</p> <p>"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)</p> <p>"O" document referring to an oral disclosure, use, exhibition or other means</p> <p>"P" document published prior to the international filing date but later than the priority date claimed</p> </div> <div style="width: 45%;"> <p>"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</p> <p>"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step</p> <p>"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.</p> <p>"&" document member of the same patent family</p> </div> </div>		
IV. CERTIFICATION		
Date of the Actual Completion of the International Search ²	Date of Mailing of this International Search Report ²	
February 1, 1984	06 FEB 1984	
International Searching Authority ¹	Signature of Authorized Officer ²⁰	
ISA/US	Eddie Chan	

FURTHER INFORMATION CONTINUED FROM THE SECOND SHEET

V. ☐ OBSERVATIONS WHERE CERTAIN CLAIMS WERE FOUND UNSEARCHABLE ¹⁰

This international search report has not been established in respect of certain claims under Article 17(2) (a) for the following reasons:

1. ☐ Claim numbers _____, because they relate to subject matter ¹² not required to be searched by this Authority, namely:

2. ☐ Claim numbers _____, because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out ¹³, specifically:

VI. ☒ OBSERVATIONS WHERE UNITY OF INVENTION IS LACKING ¹¹

This International Searching Authority found multiple inventions in this international application as follows:

Group I- Claims 1-6

See PCT/ISA 206 for the
differences.

Group II- Claims 7-16

1. ☒ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims of the international application. (See PCT/ISA 206-Authorized Additional fees)

2. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims of the international application for which fees were paid, specifically claims:

3. ☐ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claim numbers:

4. ☐ As all searchable claims could be searched without effort justifying an additional fee, the International Searching Authority did not invite payment of any additional fee.

Remark on Protest

☐ The additional search fees were accompanied by applicant's protest.

☒ No protest accompanied the payment of additional search fees.