PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 4 :		(11) International Publication Number:
G06F 11/14, H04M 3/42	A1	WO 86/ 01018
		(43) International Publication Date: 13 February 1986 (13.02.86)

(21) International Application Number: PCT/US85/01279

(22) International Filing Date: 3 July 1985 (03.07.85)

(31) Priority Application Number: 634,460

(32) Priority Date: 26 July 1984 (26.07.84)

(33) Priority Country: US

(71) Applicant: AMERICAN TELEPHONE & TELEGRAPH COMPANY [US/US]; 550 Madison Avenue, New York, NY 10022 (US).

(72) Inventor: NG, Fred, Kwok-Fai ; 1323 Sunnybrook Drive, Naperville, IL 60540 (US).

(74) Agents: HIRSCH, A., E., Jr. et al.; Post Office Box 901, Princeton, NJ 08540 (US).

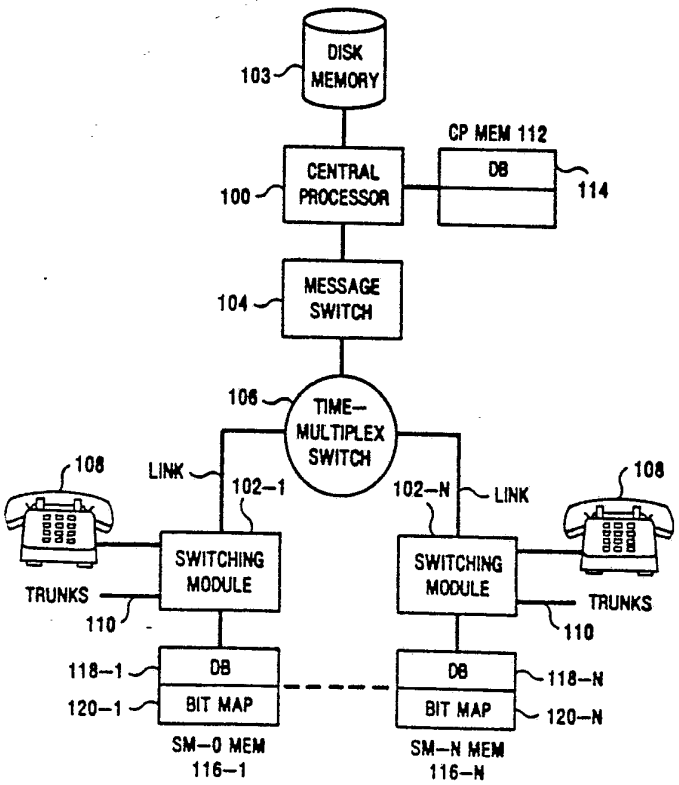
(81) Designated States: AT (European patent), BE (European patent), CH (European patent), DE (European patent), FR (European patent), GB (European patent), IT (European patent), JP, KR, LU (European patent), NL (European patent), SE (European patent).

Published
With international search report.

(54) Title: DATABASE BACKUP METHOD

(57) Abstract

A method of generating a backup copy of a database system. Modifications made to the database since the generation of a prior backup copy are summarized in a bit map on a page basis. When the next backup copy is made, only modified pages are transmitted and merged with the prior copy. Plural backup passes are made. New modifications are allowed to occur to the database on all but the last pass. Any database modification made at an address that has already been examined during a pass is backed-up during the next pass. Modifications are locked out at the beginning of the last pass to allow the final generation of a consistent and complete backup copy.



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

DATABASE BACKUP METHOD

This invention relates to a method of generating a backup copy of the contents of each of the peripheral data base memories in the central computer mass memory, in a distributed data base system having a central computer, a central computer main memory, a central computer mass memory, and a plurality of peripheral data base memories.

Background of the Invention

There is a need in virtually all database systems to periodically produce a backup copy, or dump, of the contents of the stored information in the system. This is because the stored information in databases typically changes over time. A backup copy of the contents of a database at some point in time assures that the database can be restored to the state existing at that point in time in case of a malfunction that corrupts the database.

The classic method of producing a database backup is to establish a system state in which modifications to the database are disallowed and then to copy the entire database information directly onto a mass memory, such as a backup disc or tape. Unfortunately, in many types of databases this method incurs a large expense in terms of system unavailability to users. For example, according to an article by Kaunitz and Ekert appearing in the Australian Computer Journal, Vol. 13, No. 4, November 1981, entitled "Database Backup-The Problem of Very Large Databases," the time and resources required to backup databases larger than one billion bytes are viewed as critical problems by the database users.

A similar problem can occur in a small distributed database system if a backup of the distributed database is to be produced on a single resource, such as a disk. If the communication channels between the distributed database are slow or are congested, the real time required to take a full backup of each database site may approach that of a large centralized database. Thus, the disadvantages

inherent in the classic method of dumping may be present.

The above article by Kaunitz and Ekert surveys several techniques which have been proposed to alleviate problems of database backup. According to the article, some systems provide an elaborate dynamic backup mechanism whereby database backups are made entirely in the presence of normal user processing, including ongoing modification of the database information. In these techniques, since a backup completed at any given time usually does not represent a consistent state of the database, other means must be used to place the database reloaded from a backup file in a logically consistent state. A typical method of doing this is to maintain an audit trail of modifications to a database as the modifications occur. Having reloaded the database from a backup file, the audit trail must be reprocessed from the earliest point in time reflected by the backup file to the time that the database was corrupted. Then the effect of modifications that were in progress at the time of failure must be removed by some process. Obviously, this is an expensive and complicated process.

Kaunitz and Ekert also mention a differential backup technique as an alternative to the classic backup technique. The database is viewed as consisting of a number of pages of fixed size. A full backup is initially taken of the database. Thereafter, as modifications are made to the database, bit indications are entered into a bit map according to the pages that are affected. Eventually a backup differential file is created by saving only those pages that have been modified as indicated by the bit map. Periodically, this differential backup file is updated with the pages that changed since the last differential backup. At some point, the differential backup file is either merged with the full backup file or a new full backup is taken.

The differential method above also has certain problems. First, a separate differential backup file must

be maintained in addition to a full backup file. In the event that the database must be rebuilt, the full backup must be reloaded and then merged with the differential file. Secondly, the system is unavailable for updates during a backup process.

Summary of the Invention

The problems are solved in accordance with this invention by a method of generating a backup copy of the contents of each of the peripheral data base memories in which the method comprises the steps of sequentially executing a first update pass on each of the peripheral memories one or more times, wherein the first pass includes establishing a state in which update modifications are allowed to the peripheral memory being backed-up at a given time during execution of the first pass, and sequentially transmitting all modifications of the peripheral memory being backed-up at a given time to the central computer and thence to the mass memory, sequentially executing a second update pass on each of the peripheral memories, wherein the second pass includes establishing a state in which update modifications to the peripheral memory being updated at a given time are not allowed, and sequentially transmitting all additional modifications made to the peripheral memory being backed-up since execution of the last first pass to the central computer and thence to the mass memory.

The preferred and disclosed embodiment of the invention is in a distributed database system involving slow communication channels between each of the individual databases and a central processor. This system may be viewed as equivalent, in terms of the time required to generate a backup, to a large database in which the backup time may be a problem. Two passes are made in the preferred embodiment. The individual databases are sequentially backuped during each pass. The individual database being backuped at any given time is locked to modifications during the last pass. This assures that virtually the distributed database as a whole is available

to users during almost the entire time required to generate the backup.

Brief Description of the Drawing

In the drawing,

Fig. 1 is a block diagram of a distributed computer-controlled telephone system, including a central processor and one or more switching modules, used to practice the invention;

Fig. 2 recited with Fig. 6 illustrates how a database memory associated with a switching module is divided into pages and how a bit map is used to record pages of the memory that have been modified since the last generation of a backup copy of the memory;

Fig. 3 shows an illustrative flowchart of the main backup program used to control the central processor during backup generation,

Fig. 4 shows illustrative flowchart details of a backup subroutine called by the main program of Fig. 3;

Fig. 5 shows illustrative flowchart details of a backup program located in memory at each of the switch modules; and

Fig. 6 shows an illustrative format of a differential backup message sent by a switch module to the central processor during a backup process.

Detailed Description

Fig. 1 shows a simplified block diagram of a distributed number 5 electronic switching telephone system. This system is used to illustrate the principles of the invention. Fundamentally, the system comprises a central processor 100 and one or more switching modules 102-1 to 102-N. In the illustrative system, central processor 100 is a commercially available processor. It handles the functions that require centralized intelligence, including allocation of resources, overall maintenance, and interface with operations support systems for record keeping, troubleshooting, and so forth.

The central processor 100 is connected to the

switch modules, a message switch 104 and a time-multiplexed switch 106. Message switch 104 directs the routing of control messages between central processor 100 and the switch modules and between the switch modules themselves. These messages travel through the time-multiplex switch 106.

The switch models 102 are microprocessor controlled units that provide most call processing functions. They provide the interfaces between the system and stations and trunks, 108 and 110. Each switch module contains a time slot interchange unit that performs the first stage of switching. Connections involving more than a single switch module are provided by the time-multiplex switch 106.

Central processor 100 is associated with a random access memory 112, part of which is a CP database 114. This database contains such items as office dependent data and customer data that requires centralized access for functions performed by CP 100. Each of the switch modules 102-1 through 102-N is also associated with a separate random access memory 116-1 through 116-N, part of which is a switch module database (SMDB) 118-1 through 118-N. The SMDBs typically contain office dependent data individual to the SM, such as station and trunk appearances, and feature data specific to the stations served by the SM. The CP database 114 and the SM databases 118 make up a distributed database of the system. CP 100 is also associated with a disk memory 103 which, among other things, is used to store a most recent backup copy of the distributed database.

Another part of each SM memory 116 is a bit map 120. A SM database 118 is conceptually divided into pages of an arbitrary and fixed size. In the illustrative embodiment, a page is 128 bytes in length. When a modification is made to a SMDB 118, such as activation of a customer calling feature, a "1" is entered into the appropriate bit map at a bit location defining the page

containing the modification. This is illustrated in Fig. 2 in which a bit map having words 0 to P is shown, each assumed to have 10 bits each. To the right of the bit map is a page layout of a SMDB. As shown, the first bit of the bit map is associated with page 0 of the database. The right-most bit of word 0 is associated with page 9 and is set to "1", indicating that page 9 has been modified since a last backup copy was taken.

The program control functions described below have been simplified in the interest of brevity and understanding. For example, the flowcharts of Figs. 3 through 5 are drawn without concern for such things as real time breaks in program flow, which are commonplace in the industry and within the skill of the ordinary art worker.

Database backup may be scheduled to execute periodically, say once a day, or to be executed on demand from, say, a system operator. In either event, entry is made to CP 100 program address START in Fig. 3 to begin the backup. START is a backup control program. Fundamentally, it controls the successive execution of a backup subroutine shown in Fig. 4 for each SM 102 in an office. Each time the backup subroutine is completed on all the SMs of an office, a pass is said to be completed. At least two and as many passes as desired may be performed. In accordance with the invention, modifications are allowed to occur in the SM databases 118 while backup is being performed during all passes except the last pass. The result is that during the initial passes in which the greater number of modifications are expected to be transmitted to CP 100, and thence to disk 103, the SMDBs 118 are still available for modification. During the final pass, SM database modification lockout occurs for only the very short period of time required to transmit the few, if any, modifications that have occurred to a SM database since the immediately preceding pass. Since modified pages are transmitted from each SMDB in sequential order, modifications that occur to a SM database, such as 118-1, at a page location that has

already been examined during a pass on the database are backed-up during the next pass on the database.

Three backup passes 1 through 3 are shown in Fig. 3 for illustration, although it has been found that in the exemplary system two passes reduce the modification lockout time to a satisfactory value. In Fig. 3, step 300 sets a pass indicator to pass 1. Steps 302 and 304 call the backup subroutine BACKUP sequentially for each SM of an office until the pass is executed on all SMs. At this point, steps 306, 308 and 310 execute pass 2 on all the SMs to backup intervening modifications to the SM databases that arrived too late to be included during pass 1. In accordance with the invention, it is normally expected that the number of modifications backed-up by step 310 is much less than those backed-up by step 304.

After step 310 is executed on all SMs, step 312 sets the pass indicator to pass 3. Step 313 sets an indicator in the CP memory 112 to lockout modifications to the CP database 114. This is done because some CP database modifications cause related modifications to occur in the SM databases. Because modifications are not allowed to an SM that is undergoing a pass 3 backup, modifications must also be disallowed to the CP database.

Steps 314 and 316 now execute the backup subroutine sequentially on each of the SMs 102. When the last SM is completed, step 318 backs up the CP database 114 by copying the entire CP database to disk 103. Since there is no slow communication channel between the CP database and CP 100 and because there is only one CP database, there is no need to backup the CP database in passes. The speed of generating the CP database backup in disk 103 is sufficient to not lock out modifications for an unduly long period of time in this embodiment. However, it is noted that the CP backup could be generated in passes to avoid such a problem if circumstances were to so warrant.

After backup of the CP database is complete, step 320 unlocks the database by resetting the CP lock

indication in memory 112. At this point backup is complete. A copy of the CP database as it exists at this point in time and a copy of each of the SM databases as they existed at the time of completion of pass 3 on each individual database exists in disk 103.

Fig. 4 shows the details of the BACKUP subroutine that controls the operations of CP 100. Step 400 determines if pass 3 is in progress. If not, update lockup is not in effect and step 402 causes CP 100 to send a backup alert message to the SM on which this pass is being executed. This alert message includes an "unlocked" flag for passes 1 and 2. On the other hand, if pass 3 is in effect, step 404 sends the alert message to the SM, including a "lock" flag. Fig. 5 shows the details of the program that controls the microprocessors in SMs 102 in response to control messages from CP 100. Step 504 in the SM 102 being addressed recognizes the alert message from CP 100. Step 506 determines if the alert message includes a request to lockout database modifications. If so, step 508 places the lockout state into effect. In either event, step 510 next returns an acknowledge (ACK) message to CP 100 that all is ready to begin the pass.

Step 406 in the CP 100 program recognizes receipt of the ACK message resulting in step 408 transmitting a backup initiation message to the SM. Referring again to Fig. 5, step 502 recognizes the initiation message. In response, step 512 performs certain minor initialization functions such as setting a bit map search pointer to start at the beginning of the bit map. Step 514 begins a search of the bit map shown in Fig. 2 to locate the next "1" therein indicating a page modification. If no "1" is found, steps 516 and 530 result in the generation and transmission of a message to CP 100 with the "finished" flag set. Step 534 looks for a responsive ACK message from CP 100 and terminates SM START program operation when it is received. On the other hand, assuming that a first "1" is found, step 520 translates the position of the "1" in the

bit map into a page address in the appropriate SM database 118. Step 522 clears the "1" in the bit map. Step 528 transmits the page associated with the first "1" to CP 100 with the finished flag set to "0" (not finished). Step 532 waits for an acknowledge (ACK) message from CP 100 and continues at step 514 when the ACK message is received to locate the next "1" in the bit map. This process continues until all modified pages as indicated in the bit map are transmitted to CP 100.

The above description of the SM backup program is somewhat simplified from what has been found to be preferred. In the specific system described in Fig. 1, it is efficient to transmit up to five modified pages at a time to CP 100 without waiting for an acknowledge (ACK) message from the CP, rather than one page at a time as illustrated in Fig. 5. This, however, is expected to vary with particular system environments and an appropriate revision of Fig. 5 along these lines is well within the state of art of skilled art workers.

With reference again to the CP 100 program shown in Fig. 4, step 410 receives each transmission of a modified page from an SM then active in a pass. Step 411 returns an acknowledgement (ACK) message to the SM in response to the ACK message. Step 412 tests the "finished" flag in the message to determine if backup is complete on this pass for this SM. If the flag is not set, the message contains a page of data. Step 414 merges the page into the previous backup copy which exists on disk 103. The process then continues on this SM at step 410. Otherwise, step 424 transmits a database update unlock message to the SM just completed. This message is decoded at step 500 of the SM program in Fig. 5, causing step 501 to unlock modifications to the SM database. This step actually is effective only for the last pass (pass 3), since updates are already unlocked for the earlier passes.

Referring again to Fig. 4, at step 426 the CP 100 backup subroutine returns to the appropriate point at

step 304, 310 or 316 in the main program after transmitting the unlock message.

In view of the above teaching it is seen that the SM databases 118 are available for updates by users virtually the entire time of backup. Even during the last pass (pass 3), only the SM database actively being backed at any time is locked to updates. Moreover, the differential backup, i.e., the transmission only of modified pages and the merging of the differential updates into the permanent backup disk copy greatly decreases the amount of data required to be processed over the classic full copy method.

It is to be understood that the above-described arrangement is merely illustrative of the application of the principles of the invention and that other arrangements may be devised by those skilled in the art without departing from the spirit and scope of the invention.

Claims

1. A method of generating a backup copy of the contents of each of the peripheral data base memories in the central computer mass memory, in a distributed data base system having a central computer, a central computer main memory, a central computer mass memory, and a plurality of peripheral data base memories, comprising the steps of

sequentially executing a first update pass on each of the peripheral memories one or more times, wherein the first pass includes

establishing a state in which update modifications are allowed to the peripheral memory being backed-up at a given time during execution of the first pass, and

sequentially transmitting all modifications of the peripheral memory being backed-up at a given time to the central computer and thence to the mass memory,

sequentially executing a second update pass on each of the peripheral memories, wherein the second pass includes

establishing a state in which update modifications to the peripheral memory being updated at a given time are not allowed, and

sequentially transmitting all additional modifications made to the peripheral memory being backed-up since execution of the last first pass to the central computer and thence to the mass memory.

2. The method of claim 1 wherein each of the peripheral memories is partitioned into pages of a prescribed and the method further comprises

setting a bit in a bit map contained in a said peripheral memory and corresponding to a page in the peripheral memory to a prescribed state to indicate a modification to that page.

3. The method of claim 2 wherein the steps of transmitting modifications during the first and second passes further comprises

sequentially scanning the bit map for the next bit

set to the first prescribed state,

setting the next bit found to a second prescribed state, and

transmitting the contents of the page of the peripheral memory identified by the bit to the central computer.

4. The method of claim 3 wherein the step of sequentially scanning the bit map further comprises

resuming the sequential scanning of the bit map at the immediate next bit address after having located a previous bit set to the first prescribed state and transmitting the corresponding page to the central computer.

FIG. 1

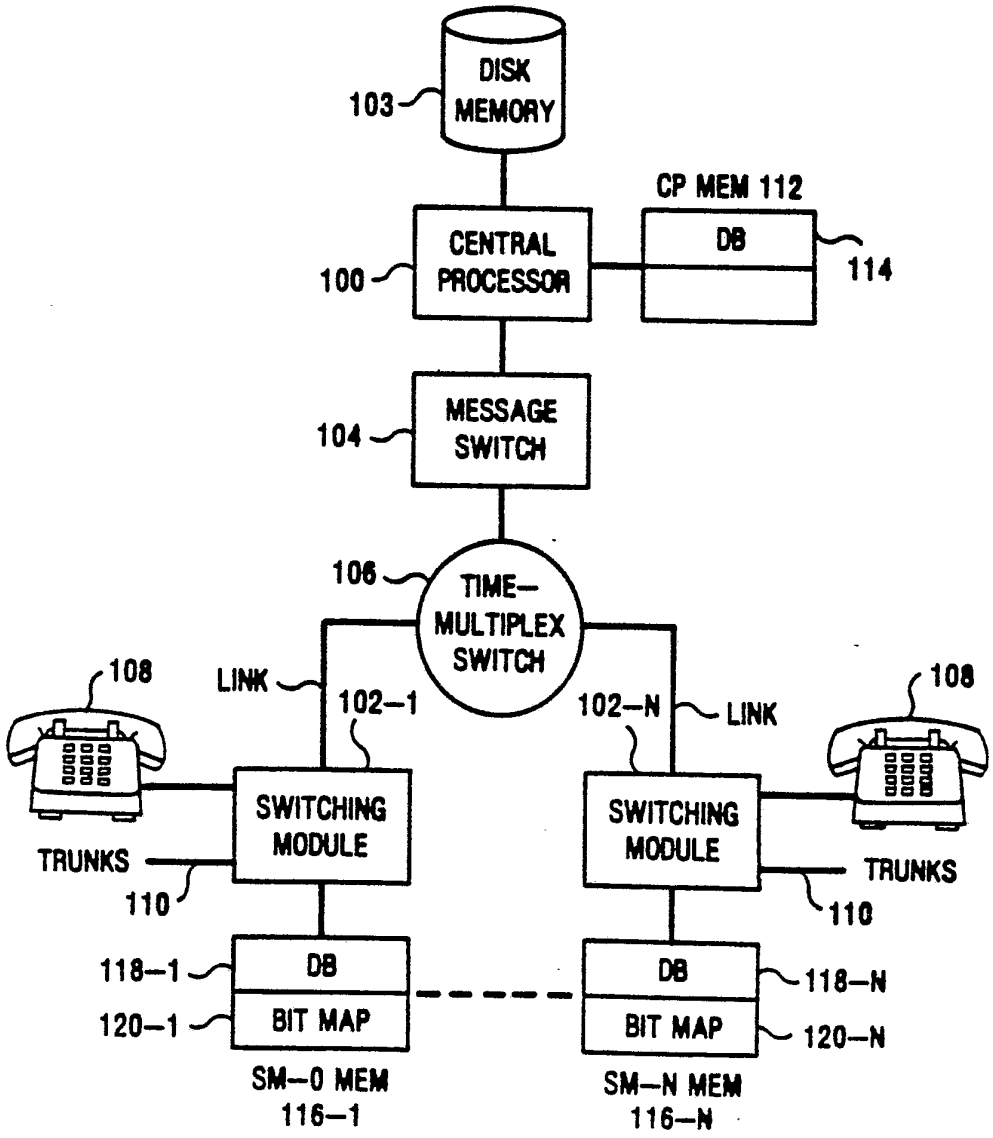

DISK MEMORY
103
CP MEM 112
DB
114
CENTRAL PROCESSOR
100
MESSAGE SWITCH
104
106
TIME-MULTIPLEX SWITCH
LINK
102-1
102-N
LINK
108
108
SWITCHING MODULE
SWITCHING MODULE
TRUNKS
110
TRUNKS
110
118-1
DB
BIT MAP
120-1
SM-0 MEM
116-1
DB
118-N
BIT MAP
120-N
SM-N MEM
116-N

FIG. 2

SM MEM

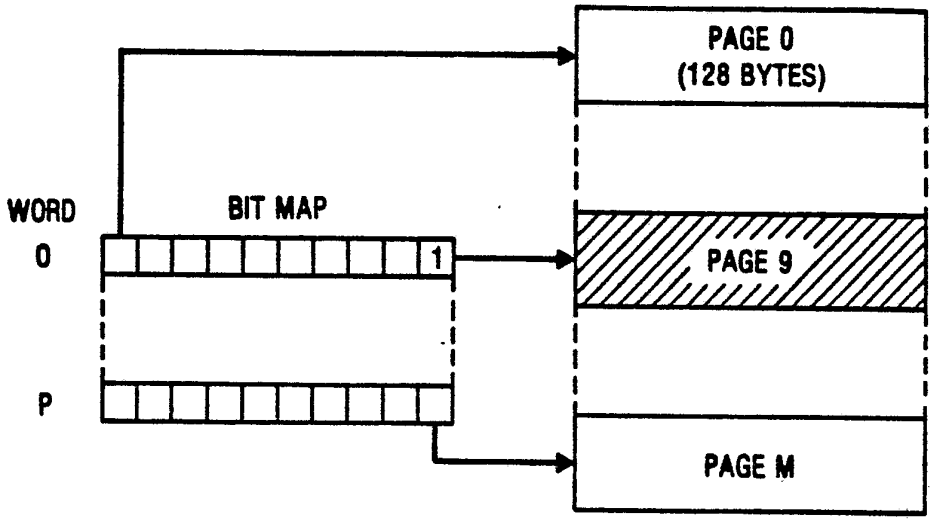


FIG. 6

DIFFERENTIAL UPDATE MSG FROM SM

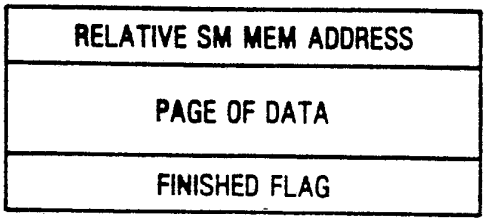


FIG. 3

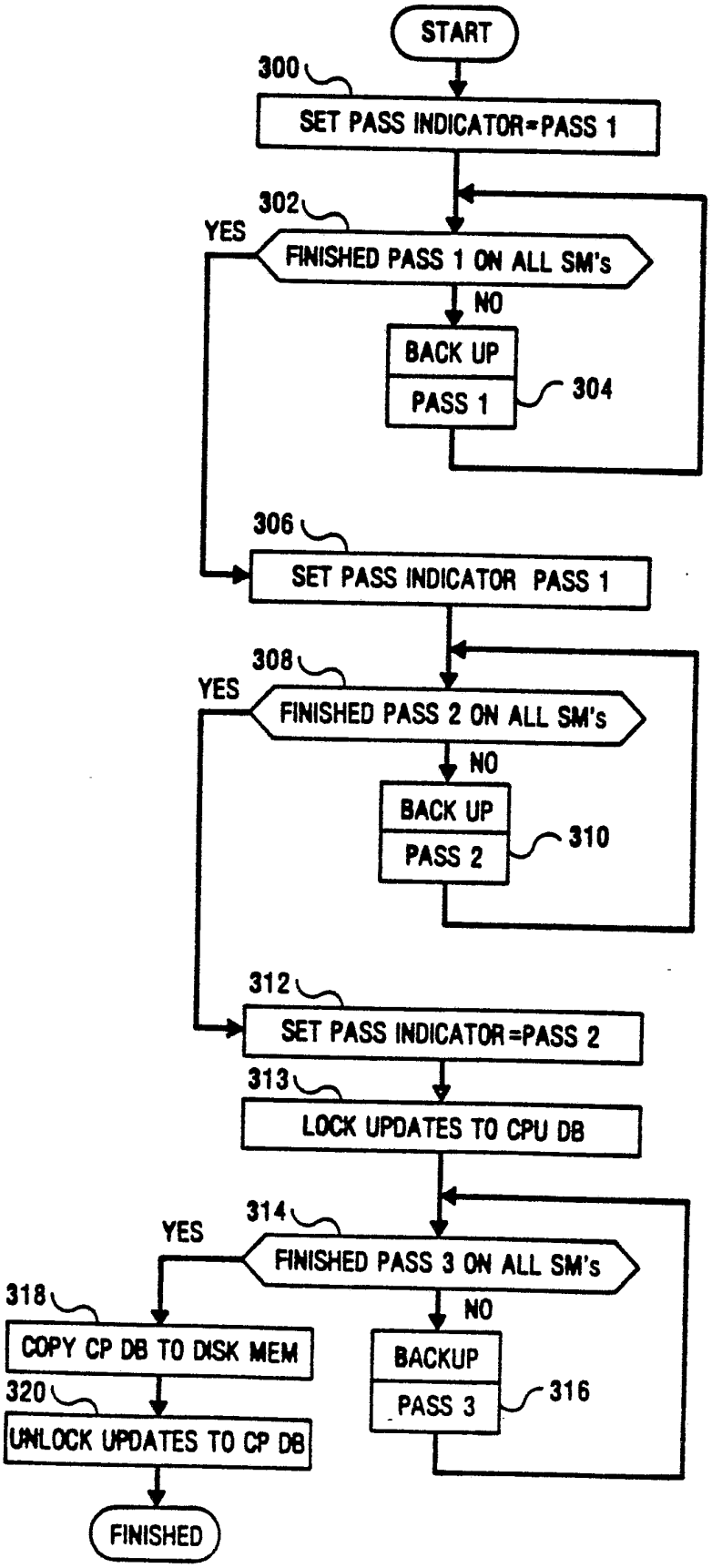

START
300
SET PASS INDICATOR=PASS 1
302
YES
FINISHED PASS 1 ON ALL SM's
NO
BACK UP
PASS 1
304
306
SET PASS INDICATOR PASS 1
308
YES
FINISHED PASS 2 ON ALL SM's
NO
BACK UP
PASS 2
310
312
SET PASS INDICATOR=PASS 2
313
LOCK UPDATES TO CPU DB
314
YES
FINISHED PASS 3 ON ALL SM's
NO
318
COPY CP DB TO DISK MEM
BACKUP
PASS 3
316
320
UNLOCK UPDATES TO CP DB
FINISHED

FIG. 4

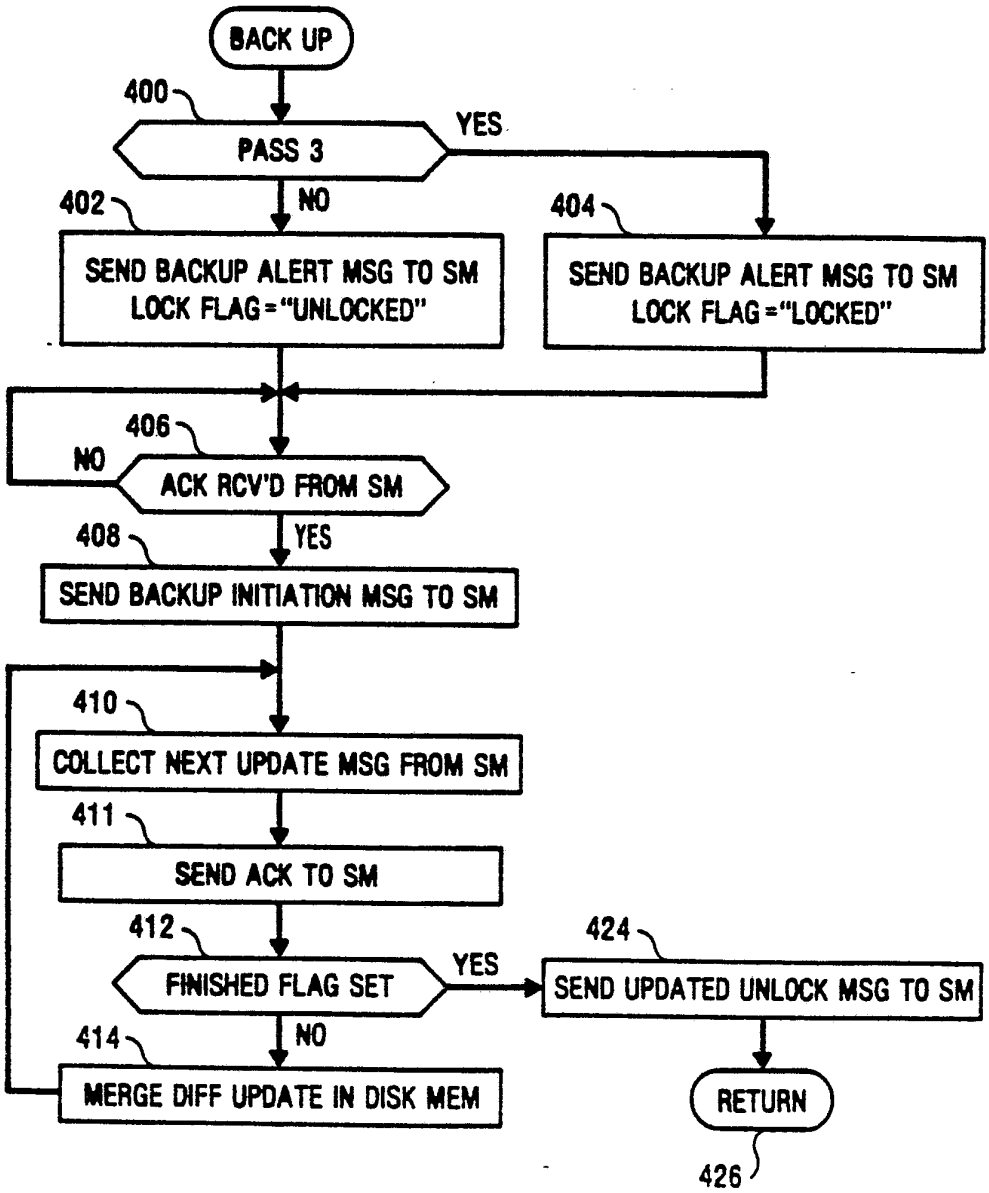
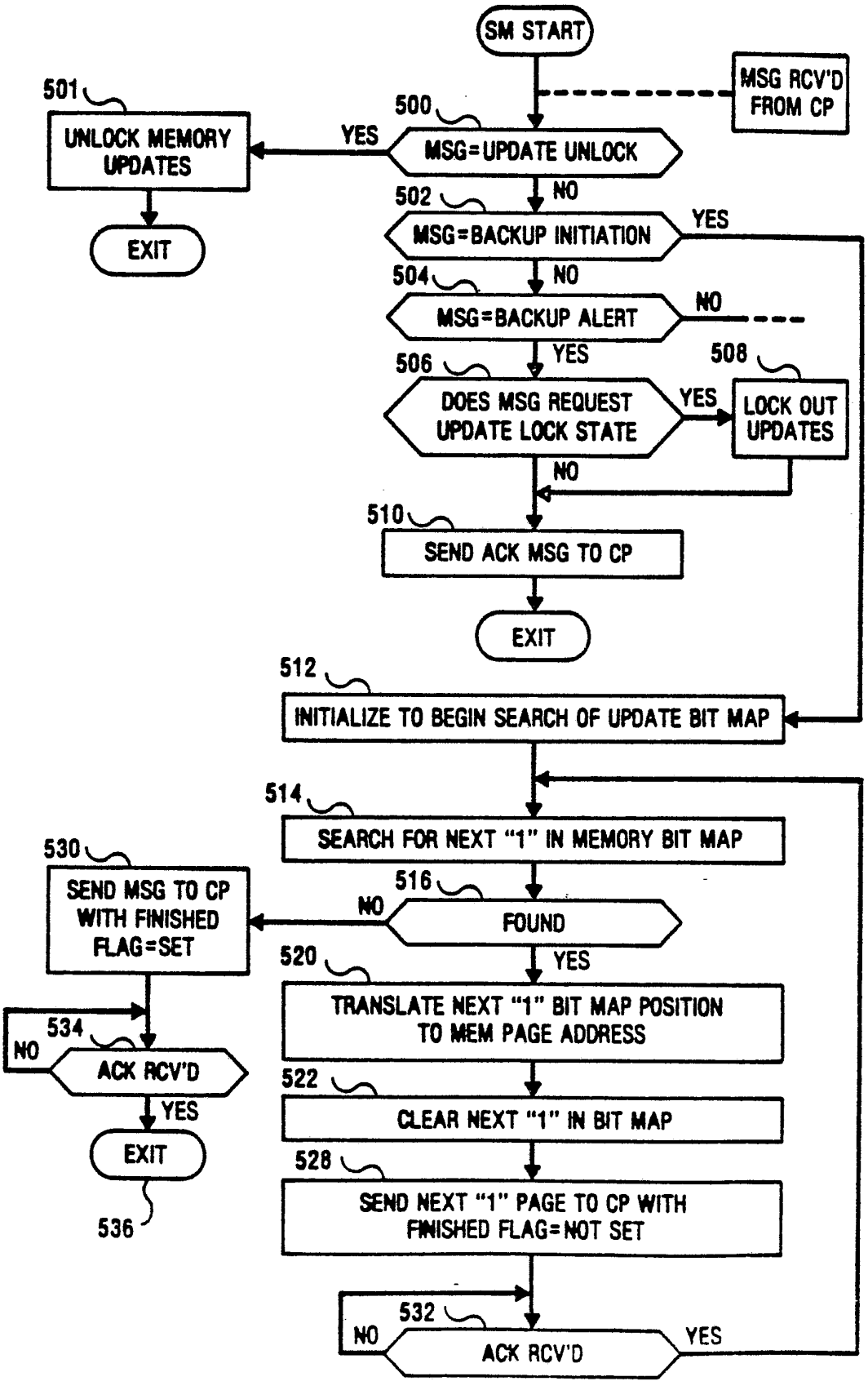

BACK UP
400
PASS 3
YES
NO
402
SEND BACKUP ALERT MSG TO SM
LOCK FLAG = "UNLOCKED"
404
SEND BACKUP ALERT MSG TO SM
LOCK FLAG = "LOCKED"
406
NO
ACK RCV'D FROM SM
YES
408
SEND BACKUP INITIATION MSG TO SM
410
COLLECT NEXT UPDATE MSG FROM SM
411
SEND ACK TO SM
412
FINISHED FLAG SET
YES
424
SEND UPDATED UNLOCK MSG TO SM
NO
414
MERGE DIFF UPDATE IN DISK MEM
RETURN
426

FIG. 5


SM START
MSG RCV'D FROM CP
500
MSG=UPDATE UNLOCK
YES
501
UNLOCK MEMORY UPDATES
EXIT
NO
502
MSG=BACKUP INITIATION
YES
NO
504
MSG=BACKUP ALERT
NO
YES
506
DOES MSG REQUEST UPDATE LOCK STATE
YES
508
LOCK OUT UPDATES
NO
510
SEND ACK MSG TO CP
EXIT
512
INITIALIZE TO BEGIN SEARCH OF UPDATE BIT MAP
514
SEARCH FOR NEXT "1" IN MEMORY BIT MAP
516
NO
FOUND
YES
530
SEND MSG TO CP WITH FINISHED FLAG=SET
520
TRANSLATE NEXT "1" BIT MAP POSITION TO MEM PAGE ADDRESS
522
CLEAR NEXT "1" IN BIT MAP
528
SEND NEXT "1" PAGE TO CP WITH FINISHED FLAG=NOT SET
534
NO
ACK RCV'D
YES
EXIT
536
532
NO
ACK RCV'D
YES

INTERNATIONAL SEARCH REPORT

International Application No PCT/US 85/01279

I. CLASSIFICATION OF SUBJECT MATTER (if several classification symbols apply, indicate all) [6]

According to International Patent Classification (IPC) or to both National Classification and IPC

IPC[4]: G 06 F 11/14; H 04 M 3/42

II. FIELDS SEARCHED

Minimum Documentation Searched [7]

Classification System	Classification Symbols
IPC[4]	G 06 F 11/14 G 06 F 15/40 H 04 M 3/42

Documentation Searched other than Minimum Documentation to the Extent that such Documents are Included in the Fields Searched [8]

III. DOCUMENTS CONSIDERED TO BE RELEVANT [9]

Category *	Citation of Document, [11] with indication, where appropriate, of the relevant passages [12]	Relevant to Claim No. [13]
A	The Australian Computer Journal, volume 13, nr. 4, November 1981, Kaunitz et al.:"Data Base Backup- The Problem of very large Data Bases", pages 136-142, see page 136, column 1, lines 29-39, column 2, lines 1-6; page 138, column 1, lines 33-39, column 2, lines 5-20,43-65; page 139, column 1, lines 1-27, column 2, lines 11-16 (cited in the application)	1-4
A	Journal of Telecommunication Networks, volume 2, nr. 3, 1983, Rockville, (US) Bray:"Distributed Database Management: Concepts and Administration", pages 237-248, see page 243, column 1, lines 15-24; page 244, column 1, lines 17-23, column 2, lines 1-25	1
A	US, A, 3760364 (YAMAUCHI) 18 September 1973, see column 2, lines 12-21; column 6, lines 27-52	1
A	Conference Record, volume 1 3, IEEE International Conference on Communications	

* Special categories of cited documents: [10]

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

IV. CERTIFICATION

Date of the Actual Completion of the International Search	Date of Mailing of this International Search Report
4th January 1985	12 NOV. 1985
International Searching Authority	**Signature of Authorized Officer**
EUROPEAN PATENT OFFICE	M. DE RICO

III. DOCUMENTS CONSIDERED TO BE RELEVANT (CONTINUED FROM THE SECOND SHEET)

Category *	Citation of Document, with indication, where appropriate, of the relevant passages	Relevant to Claim No
	ICC '82, New York, (US) Cohen et al.:"Distributed Database Management to support Network Services", pages 1.F.5.1/1.F.5.4 see page 1F.5.2, column 2, lines 34-47; page 1F.5.3; column 1, lines 27-35, column 2, lines 1-9	1

ANNEX TO THE INTERNATIONAL SEARCH REPORT ON

INTERNATIONAL APPLICATION NO. PCT/US 85/01279 (SA 10183)

This Annex lists the patent family members relating to the patent documents cited in the above-mentioned international search report. The members are as contained in the European Patent Office EDP file on 30/10/85

The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

Patent document cited in search report	Publication date	Patent family member(s)		Publication date
US-A- 3760364	18/09/73	FR-A-	2112513	16/06/72
		DE-A,B,C	2155102	25/05/72
		GB-A-	1353526	22/05/74
		BE-A-	774981	01/03/72