(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
25 July 2013 (25.07.2013)

WIPO | PCT

(10) International Publication Number
WO 2013/108097 A1

(51) International Patent Classification:
G06F 12/08 (2006.01)

(21) International Application Number:
PCT/IB2012/057140

(22) International Filing Date:
10 December 2012 (10.12.2012)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
13/352,230 17 January 2012 (17.01.2012) US

(71) Applicant: **INTERNATIONAL BUSINESS MACHINES CORPORATION** [US/US]; New Orchard Road, Armonk, New York 10504 (US).

(71) Applicants *(for MG only)*: **IBM UNITED KINGDOM LIMITED** [GB/GB]; PO Box 41, North Harbour, Portsmouth Hampshire PO6 3AU (GB). **IBM (CHINA) INVESTMENT COMPANY LIMITED** [CN/CN]; 25/F, Pangu Plaza, No.27, Central North 4th Ring Road, Chaoyang District, Beijing 100101 (CN).

(72) Inventors: **GUPTA, Lokesh, Mohan**; Ibm Corporation, Md:9032-1 103, 9000 S Rita Rd, Tucson, Arizona 85744-0002 (US). **KALOS, Matthew, Joseph**; Ibm Corporation, Md: 9032-1 102, 9000 S Rita Rd, Tucson, Arizona 85744-0002 (US). **BENHASE, Michael, Thomas**; Ibm Corporation, Md: 9032-1 102, 9000 S Rita Rd, Tucson, Arizona 85744-0002 (US). **NIELSEN, Karl, Allen**; Ibm Corporation, Md: 9032-1 103, 9000 S Rita Rd, Tucson, Arizona 85744-0002 (US). **ASH, Kevin, John**; Ibm Corporation, Md: 9032-1 103, 9000 S Rita Rd, Tucson, Arizona 85744-0002 (US).

(74) Agent: **GASCOYNE, Belinda**; IBM United Kingdom Limited, Intellectual Property Law, Hursley Park, Winchester Hampshire SO21 2JN (GB).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM,

[Continued on next page]

(54) Title: POPULATING A FIRST STRIDE OF TRACKS FROM A FIRST CACHE TO WRITE TO A SECOND STRIDE IN A SECOND CACHE

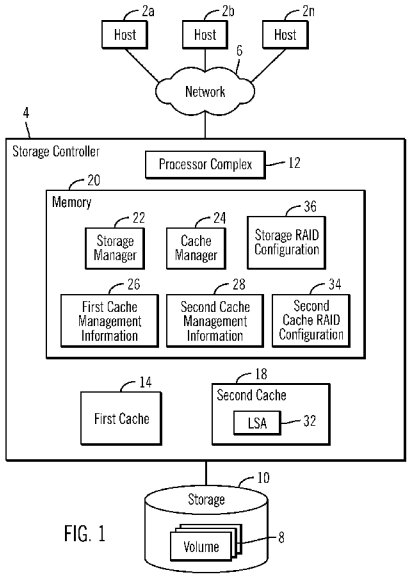


FIG. 1

(57) Abstract: Provided are a computer program product, system, and method for managing data in a cache system comprising a first cache, a second cache, and a storage system. A determination is made of tracks stored in the storage system to demote from the first cache. A first stride is formed including the determined tracks to demote. A determination is made of a second stride in the second cache in which to include the tracks in the first stride. The tracks from the first stride are added to the second stride in the second cache. A determination is made of tracks in strides in the second cache to demote from the second cache. The determined tracks to demote from the second cache are demoted.

WO 2013/108097 A1

TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

POPULATING A FIRST STRIDE OF TRACKS FROM A FIRST CACHE TO WRITE TO A SECOND STRIDE IN A SECOND CACHE

BACKGROUND OF THE INVENTION

Field of the Invention

[0001] The present invention relates to a computer program product, system, and method for populating a first stride of tracks from a first cache to write to a second stride in a second cache.

Description of the Related Art

[0002] A cache management system buffers tracks in a storage device recently accessed as a result of read and write operations in a faster access storage device, such as memory, than the storage device storing the requested tracks. Subsequent read requests to tracks in the faster access cache memory are returned at a faster rate than returning the requested tracks from the slower access storage, thus reducing read latency. The cache management system may also return complete to a write request when the modified track directed to the storage device is written to the cache memory and before the modified track is written out to the storage device, such as a hard disk drive. The write latency to the storage device is typically significantly longer than the latency to write to a cache memory. Thus, using cache also reduces write latency.

[0003] A cache management system may maintain a linked list having one entry for each track stored in the cache, which may comprise write data buffered in cache before writing to the storage device or read data. In the commonly used Least Recently Used (LRU) cache technique, if a track in the cache is accessed, i.e., a cache "hit", then the entry in the LRU list for the accessed track is moved to a Most Recently Used (MRU) end of the list. If the requested track is not in the cache, i.e., a cache miss, then the track in the cache whose entry is at the LRU end of the list may be removed (or destaged back to storage) and an entry for the track data staged into cache from the storage is added to the MRU end of the LRU list.

With this LRU cache technique, tracks that are more frequently accessed are likely to remain in cache, while data less frequently accessed will more likely be removed from the LRU end of the list to make room in cache for newly accessed tracks.

[0004] There is a need in the art for improved techniques for using cache in a storage system.

SUMMARY

[0005] Provided are a computer program product, system, and method for managing data in a cache system comprising a first cache, a second cache, and a storage system. A determination is made of tracks stored in the storage system to demote from the first cache. A first stride is formed including the determined tracks to demote. A determination is made of a second stride in the second cache in which to include the tracks in the first stride. The tracks from the first stride are added to the second stride in the second cache. A determination is made of tracks in strides in the second cache to demote from the second cache. The determined tracks to demote from the second cache are demoted.

BRIEF DESCRIPTION OF THE DRAWINGS

[0006] FIG. 1 illustrates an embodiment of a computing environment.

[0007] FIG. 2 illustrates an embodiment of first cache management information.

[0008] FIG. 3 illustrates an embodiment of second cache management information.

[0009] FIG. 4 illustrates an embodiment of a first cache control block.

[0010] FIG. 5 illustrates an embodiment of a second cache control block.

[0011] FIG. 6 illustrates an embodiment of stride information.

[0012] FIG. 7 illustrates an embodiment of a second cache RAID configuration.

[0013] FIG. 8 illustrates an embodiment of a storage RAID configuration.

[0014] FIG. 9 illustrates an embodiment of operations to demote unmodified non-sequential tracks from the first cache to promote to the second cache.

[0015] FIG. 10 illustrates an embodiment of operations to add a track to the first cache.

[0016] FIG. 11 illustrates an embodiment of operations to add tracks from the first stride to the second stride.

[0017] FIG. 12 illustrates an embodiment of operations to free space in the second cache.

[0018] FIG. 13 illustrates an embodiment of operations to free strides in the second cache.

[0019] FIG. 14 illustrates an embodiment of operations to process a request for tracks to return to a read request.

DETAILED DESCRIPTION

[0020] Described embodiments provide techniques for promoting tracks from a first cache in strides so that the tracks may be written as full stride writes to strides in the second cache to improve the efficiency of cache promotion operations. Further, while tracks are being promoted from the first cache 14 to the second cache 18 as strides, tracks are demoted from the second cache 18 on a track basis according to a cache demotion algorithm, such as an LRU algorithm. Further, strides in the second cache that are partially full, i.e., having valid and invalid tracks, may be combined into one stride to free strides in the second cache to receive further strides of tracks from the first cache, so that the second cache maintains free strides available for strides formed from the tracks in the first cache.

[0021] FIG. 1 illustrates an embodiment of a computing environment. A plurality of hosts 2a, 2b...2n may submit Input/Output (I/O) requests to a storage controller 4 over a network 6 to access data at volumes 8 (e.g., Logical Unit Numbers, Logical Devices, Logical Subsystems, etc.) in a storage 10. The storage controller 4 includes a processor complex 12, including one or more processors with single or multiple cores, a first cache 14 and a second cache 18. The first 14 and second 18 caches cache data transferred between the hosts 2a, 2b...2n and the storage 10.

[0022] The storage controller 4 has a memory 20 that includes a storage manager 22 for managing the transfer of tracks transferred between the hosts 2a, 2b...2n and the storage 10 and a cache manager 24 that manages data transferred between the hosts 2a, 2b...2n and the storage 10 in the first cache 14, and the second cache 18. A track may comprise any unit of data configured in the storage 10, such as a track, Logical Block Address (LBA), etc., which is part of a larger grouping of tracks, such as a volume, logical device, etc. The cache manager 24 maintains first cache management information 26 and second cache management information 28 to manage read (unmodified) and write (modified) tracks in the first cache 14 and the second cache 18.

[0023] The storage manager 22 and cache manager 24 are shown in FIG. 1 as program code loaded into the memory 20 and executed by the processor complex 12. Alternatively, some or all of the functions may be implemented in hardware devices in the storage controller 4, such as in Application Specific Integrated Circuits (ASICs).

[0024] The second cache 18 may store tracks in a log structured array (LSA) 32, where tracks are written in a sequential order as received, thus providing a temporal ordering of the tracks written to the second cache 18. In an LSA, later versions of tracks already present in the LSA are written at the end of the LSA 32. In alternative embodiments, the second cache 18 may store data in formats other than in an LSA.

[0025] The memory 20 further includes second cache RAID configuration information 34 providing information on a RAID configuration used to determine how to form a stride of tracks to store in the second cache 18. In one embodiment, the second cache 18 may be

comprised of a plurality of storage devices, such as separate solid state storage devices (SSDs), such that the strides formed of tracks from the first cache 14 are striped across the separate storage devices forming the second cache 18, such as flash memories. In a further embodiment, the second cache 18 may comprise a single storage device, such as one flash memory, such that the tracks are grouped in strides as defined by the second cache RAID configuration 34, but the tracks are written as strides to a single device, such as one flash memory, implementing the second cache 18. The tracks of strides configured for the second cache RAID configuration 34 may be written to the LSA 32 in the second cache 18 device. The second cache RAID configuration 34 may specify different RAID levels, e.g., levels 5, 10, etc.

[0026] The memory 20 further includes storage RAID configuration information 36 providing information on a RAID configuration used to determine how to write tracks from the first cache 14 or second cache 18, if the second cache 18 should store modified data, to the storage system 10, where the tracks in the destaged stride are striped across the storage devices, such as disk drives, in the storage system 10.

[0027] In one embodiment, the first cache 14 may comprise a Random Access Memory (RAM), such as a Dynamic Random Access Memory (DRAM), and the second cache 18 may comprise a flash memory, such as a solid state device, and the storage 10 is comprised of one or more sequential access storage devices, such as hard disk drives and magnetic tape. The storage 10 may comprise a single sequential access storage device or may comprise an array of storage devices, such as a Just a Bunch of Disks (JBOD), Direct Access Storage Device (DASD), Redundant Array of Independent Disks (RAID) array, virtualization device, etc. In one embodiment, the first cache 14 is a faster access device than the second cache 18, and the second cache 18 is a faster access device than the storage 10. Further, the first cache 14 may have a greater cost per unit of storage than the second cache 18 and the second cache 18 may have a greater cost per unit of storage than storage devices in the storage 10.

[0028] The first cache 14 may be part of the memory 20 or implemented in a separate memory device, such as a DRAM.

[0029] The network 6 may comprise a Storage Area Network (SAN), a Local Area Network (LAN), a Wide Area Network (WAN), the Internet, and Intranet, etc.

[0030] FIG. 2 illustrates an embodiment of the first cache management information 26 including a track index 50 providing an index of tracks in the first cache 14 to control blocks in a control block directory 52; an unmodified sequential LRU list 54 providing a temporal ordering of unmodified sequential tracks in the first cache 14; a modified LRU list 56 providing a temporal ordering of modified sequential and non-sequential tracks in the first cache 14; an unmodified non-sequential LRU list 58 providing a temporal ordering of unmodified non-sequential tracks in the first cache 14; and stride information 60 providing information on strides formed of unmodified non-sequential tracks in the first cache 14 to write to the second cache 18 as a full stride write.

[0031] In certain embodiments, upon determining that the first cache 18 is full, the modified LRU list 56 is used to destage modified tracks from the first cache 14 to the storage 10 so that the copy of those destaged tracks in the first cache 18.

[0032] Once a modified non-sequential track is destaged from the first cache 14 to the storage 10, then the cache manager 24 may designate that destaged tracks as an unmodified non-sequential track in the first cache 14 and add indication of the newly designated unmodified track to the unmodified non-sequential LRU list 58, from where it is eligible to be promoted to the second cache 14. The state of the destaged modified track may be changed by updating the first cache control block 104 to indicate the destaged modified non-sequential track as unmodified in field 106. Thus, unmodified non-sequential tracks in the first cache 14 may comprise read data or modified non-sequential tracks that were destaged to the storage 10 according to the modified LRU list 56. Thus, destaged modified tracks that become unmodified tracks in the LRU list 58 may be promoted to the second cache 14 to be available for subsequent read requests. In these embodiments, the second cache 14 comprises a read only cache to cache unmodified non-sequential tracks.

[0033] FIG. 3 illustrates an embodiment of the second cache management information 28 including a track index 70 providing an index of tracks in the second cache 18 to control

blocks in a control block directory 72; an unmodified list 74 providing a temporal ordering of unmodified tracks in the second cache 18; and stride information 78 providing information on strides of tracks written to the second cache 18. In one embodiment, the second cache 18 only stores unmodified, non-sequential tracks. In further embodiments, the second cache 18 may also store modified and/or sequential tracks.

[0034] All the LRU lists 54, 56, 58, and 74 may include the track IDs of tracks in the first cache 14 and the second cache 18 ordered according to when the identified track was last accessed. The LRU lists 54, 56, 58, and 74 have a most recently used (MRU) end indicating a most recently accessed track and a LRU end indicating a least recently used or accessed track. The track IDs of tracks added to the caches 14 and 18 are added to the MRU end of the LRU list and tracks demoted from the caches 14 and 18 are accessed from the LRU end. The track indexes 50 and 70 may comprise a scatter index table (SIT). Alternative type data structures may be used to provide the temporal ordering of tracks in the caches 14 and 18.

[0035] Non-sequential tracks may comprise Online Line Transaction Processing (OLTP) tracks, which often comprise small block writes that are not fully random and have some locality of reference, i.e., have a probability of being repeatedly accessed.

[0036] FIG. 4 illustrates an embodiment of a first cache control block 100 entry in the control block directory 52, including a control block identifier (ID) 102, a first cache location 104 of the physical location of the track in the first cache 14, information 106 indicating whether the track is modified or unmodified, information 108 indicating whether the track is a sequential or non-sequential access, and information 110 indicating a demote status for the track, such as no demotion, ready to demote, and demote complete.

[0037] FIG. 5 illustrates an embodiment of a second cache control block 120 entry in the second cache control block directory 72, including a control block identifier (ID) 122; an LSA location 124 where the track is located in the LSA 32; modified/unmodified info 126 indicating whether the track is modified or unmodified; and valid/invalid flag 128 indicating

whether the track is valid or invalid. A track in the second cache 18 is indicated as invalid if the track is updated in the first cache 14 or if the track is demoted from the second cache 18.

[0038] Once a modified non-sequential track is destaged from the first cache 14 to the storage 10, then the cache manager 24 may designate that destaged tracks as an unmodified non-sequential track in the first cache 14 and add indication of the newly designated unmodified track to the unmodified non-sequential LRU list 58, from where it is eligible to be promoted to the second cache 14. The state of the destaged modified track may be changed by updating the first cache control block 100 to indicate the destaged modified non-sequential track as unmodified in field 106. Thus, unmodified non-sequential tracks in the first cache 14 may comprise read data or modified non-sequential tracks that were destaged to the storage 10 according to the modified LRU list 56. Thus, destaged modified tracks that become unmodified tracks in the LRU list 58 may be promoted to the second cache 14 to be available for subsequent read requests. In these embodiments, the second cache 14 comprises a read only cache to cache unmodified non-sequential tracks.

[0039] FIG. 6 illustrates an instance 130 of the stride information 60, 78 for one stride to be formed in the second cache 18, including a stride identifier (ID) 132, tracks 134 of the storage 10 included in the stride 132, and an occupancy 136 indicating a number of valid tracks in the stride of the total number of tracks, where the tracks in the stride that are not valid are eligible for garbage collection operations.

[0040] FIG. 7 illustrates an embodiment of the second cache RAID configuration 34 that is maintained to determine how to form strides of tracks in the second cache 18 from the tracks in the first cache 14. A RAID level 140 indicates the RAID configuration to use, e.g., RAID 1, RAID 5, RAID 6, RAID 10, etc., a number of data disks (m) 142 storing tracks of user data, and a number of parity disks (p) 144 storing parity calculated from the data disks 142, where p can be one or more, indicating the number of disks for storing the calculated parity blocks. An unmodified parity optional flag 148 indicates whether parity should be calculated for unmodified non-sequential tracks in the first cache 14 being promoted to the second cache 18. This optional flag 148 allows for only including unmodified non-sequential tracks in a stride to fill the stride with only unmodified non-sequential tracks. The

stride of unmodified non-sequential tracks in the first cache 14 may be indicated in an LSA 32, where the tracks of the stride are striped across m plus p storage devices forming the second cache 18. Alternatively, the second cache 18 may comprise fewer than n devices.

[0041] FIG. 8 illustrates an embodiment of the storage RAID configuration 36 that is maintained to determine how to form strides of modified tracks in the second cache 18 to stripe across the disks of the storage 10. A RAID level 150 indicates the RAID configuration to use, a number of data disks (m) 152 storing tracks of user data, and a number of parity disks (p) 154 storing parity calculated from the data disks 152, where p can be one or more, indicating the number of disks for storing the calculated parity blocks. The stride of tracks from the second cache 18 may be striped across disks in the storage system 10.

[0042] In one embodiment, the second cache 34 and storage 36 RAID configurations may provide different parameters or have the same parameters, such as different RAID levels, data disks, parity disks, etc.

[0043] FIG. 9 illustrates an embodiment of operations performed by the cache manager 24 to demote unmodified non-sequential tracks from the first cache 14 to promote to the second cache 18, where the unmodified non-sequential tracks may be selected from the LRU end of the unmodified non-sequential LRU list 58 when space is needed. Upon initiating (at block 200) the operation to demote selected unmodified non-sequential tracks, the demote status 110 (FIG. 4) of the unmodified non-sequential tracks selected to demote is set (at block 202) to "ready". The cache manager 24 uses (at block 204) the second cache RAID configuration information 34 to form a first stride of tracks from the first cache 114 to promote to a stride in the second cache 18. For instance, forming the first stride of tracks may comprise forming a stride for a RAID configuration based on a RAID configuration defined 34 for the second cache as having n devices including m devices for storing tracks of data and at least one parity device p to store parity data calculated from the tracks of data for the m devices. Further, the first stride of tracks may be striped across n solid state storage devices without parity to form the second stride in embodiments where the second cache comprises at least n solid state storage devices.

[0044] The cache manager 24 processes (at block 206) the unmodified non-sequential LRU 58 list to determine a number of unmodified non-sequential tracks having a demote status 110 of ready in their control blocks 100. If the cache manager 24 determines (at block 208) that the number of unmodified non-sequential tracks is sufficient to form a stride, then the cache manager 24 populates (at block 210) the first stride of unmodified non-sequential tracks having a demote status 110 of ready. In one embodiment, the first stride may be populated starting from the LRU end of the unmodified non-sequential LRU list 58 and use enough tracks for the data disks in stride. If (at block 212) the RAID configuration specifies parity disks, then the cache manager 24 calculates (at block 212) parity for the unmodified non-sequential tracks included in the stride and includes parity data (for the p parity disks) in the stride. If (at block 208) there are not sufficient unmodified non-sequential tracks in the first cache 14 to fill the first stride, then control ends until there are a sufficient number of unmodified non-sequential tracks having the demote ready status available to populate the first stride.

[0045] After populating the first stride (at blocks 210 and 212), the cache manager 14 determines (at block 214) a free second stride in the second cache 18 in which to include the tracks from the first stride. The tracks from the first stride are written or striped (at block 216) as a full stride write to the second stride across the devices forming the second cache 18. Upon filling the second stride in the second cache 18 with the tracks from the first stride, the cache manager 14 indicates (at block 218) the occupancy 136 of the stride information 130 for the second stride as full. The cache manager 24 updates (at block 220) the demote status 110 for the unmodified non-sequential tracks included in the stride as demote "complete".

[0046] Although the operations of FIG. 9 are described as demoting unmodified non-sequential tracks from the first cache 14 to promote to the second cache 18, in alternative embodiments, the operations may apply to demoting different types of tracks, such as modified, sequential, etc.

[0047] With the described embodiments, the unmodified tracks from the first cache 14 are gathered and written as a stride to the second cache 18 so that one Input/Output (I/O) operation is used to transfer multiple tracks.

[0048] FIG. 10 illustrates an embodiment of operations performed by the cache manager 24 to add, i.e., promote, a track to the first cache 14, which track may comprise a write or modified track from a host 2a, 2b...2n, a non-sequential track in the second cache 18 that is subject to a read request and as a result moved to the first cache 14, or read requested data not found in either cache 14 or 18 and retrieved from the storage 10. Upon receiving (at block 250) the track to add to the first cache 14, if (at block 252) a copy of the track is already included in the first cache 14, i.e., the received track is a write, then the cache manager 24 updates (at block 254) the track in the first cache 14. If (at block 252) the tracks is not already in the cache, then the cache manager 24 creates (at block 256) a control block 100 (FIG. 4) for the track to add indicating the location 104 in the first cache 14 and whether the track is modified/unmodified 106 and sequential/non-sequential 108. This control block 100 is added to the control block directory 52 of the first cache 14. The cache manager 24 adds (at block 258) an entry to the first cache track index 50 having the track ID of track to add and an index to the created cache control block 100 in the control block directory 52. An entry is added (at block 260) to the MRU end of the LRU list 54, 56 or 58 of the track type of the track to add. If (at block 262) the track to add is a modified non-sequential track and if (at block 264) a copy of the track to add is in the second cache 18, as determined from the second cache track index 70, then the copy of the track in the second cache 18 is invalidated (at block 266), such as by setting the valid/invalid flag 128 in the cache control block 120 for the track in the second cache 18 to invalid. If (at block 306) the track to add is unmodified sequential, control ends.

[0049] FIG. 11 illustrates an embodiment of operations performed by the cache manager 24 to add tracks from the first stride from the first cache 14 to the second stride in the second cache 18. The cache manager 24 creates (at block 302) stride information 130 (FIG. 6) for the second stride indicating the tracks 134 from the first stride being added and indicating the occupancy 136 as full. For each track in the first stride being added, a loop of operations is performed at blocks 304 through 318. The cache manager 24 adds (at block

302) indication, such as the track ID, of the track being promoted to the LSA 32 in the second cache 18. If (at block 308) the track being added is already in the second cache 18, then the cache manager 24 updates (at block 310) the cache control block 120 for the track indicating the location 124 in the LSA 32, that the data is unmodified 126, and that and that the track is valid 128. If (at block 308) the track is not already in the second cache 18, then the cache manager 24 creates (at block 312) a control block 120 (FIG. 5) for the track to add indicating the track location 124 in the LSA 32 and whether the track is modified/unmodified 126. An entry is added (at block 314) to the second cache track index 70 having the track ID of the promoted track and an index to the created cache control block 120 in the control block directory 72 for the second cache 18. From block 310 or 316, the cache manager 24 indicates (at block 316) the promoted track at the MRU end of the unmodified LRU list 74, such as by adding the track ID to the MRU end.

[0050] FIG. 12 illustrates an embodiment of operations performed by the cache manager 24 to free space in the second cache 18 for new tracks to add to the second cache 18, i.e., tracks being demoted from the first cache 14. Upon initiating this operation (at block 350) the cache manager 24 determines (at block 352) unmodified tracks in the second cache 18 from the LRU end of the unmodified LRU list 74 and invalidates (at block 354) the determined unmodified tracks without destaging the invalidated unmodified tracks to the storage 10, and also removes the invalidated unmodified tracks from the unmodified LRU list 74 and indicates the track as invalid 128 in the cache control block 120 for the track. The unmodified tracks in the second cache 18 may comprise read tracks added to the first cache 14 or modified tracks destaged from the first cache 14. Further, the tracks selected by the cache manager 24 for demotion from the second cache 18 may be from different strides formed in the second cache 18. Further, strides in the second cache may include both valid and invalid tracks, where tracks are invalidated by demoting from the second cache 18 or by the track being updated in the first cache 18.

[0051] In certain embodiments, the cache manager 24 uses different track demotion algorithms to determine tracks to demote from the first cache 14 and the second cache 18 by using separate LRU lists 58 and 74 for the first 14 and second 18 caches 18, respectively, to determine the tracks to demote. The algorithms used to select tracks for demotion in the first

14 and second 18 caches may consider characteristics of the tracks in the first 14 and second 18 caches to determine tracks to demote first.

[0052] FIG. 13 illustrates an embodiment of operations performed by the cache manager 24 to free strides in the second cache 18 to make available for strides of tracks in the first cache 14. Upon initiating (at block 370) an operation to free strides in the second cache 18, the cache manager determines (at block 372) if the number of free strides, i.e., strides having an occupancy 136 of zero, is less than a free stride threshold. For instance, the cache manager 24 may ensure that there always are at least two or some other number of free strides to be available for strides formed from the first cache 14 tracks. If the number of free strides is not below the threshold, then control ends. Otherwise, if (at block 372) the number of free strides is less than the threshold, then the cache manager 24 determines (at block 374) an available stride having zero occupancy 136 and determines (at block 376) at least two strides that are partially full, i.e., having valid and invalid tracks, whose valid tracks can fit into the free stride. The cache manager 24 combines (at block 378) the valid tracks from the determined at least two partially full strides into the determined available stride. The cache manager 24 then indicates (at block 380) the at least two strides from which the tracks merged as having zero occupancy 136, so they are available to receive tracks from strides from the first cache 14.

[0053] FIG. 14 illustrates an embodiment of operations performed by the cache manager 24 to retrieve requested tracks for a read request from the caches 14 and 18 and storage 10. The storage manager 22 processing the read request may submit requests to the cache manager 24 for the requested tracks. Upon receiving (at block 450) the request for the tracks, the cache manager 24 uses (at block 454) the first cache track index 50 to determine whether all of the requested tracks are in the first cache 14. If (at block 454) all requested tracks are not in the first cache 14, then the cache manager 24 uses (at block 456) the second cache track index 70 to determine any of the requested tracks in the second cache 18 not in the first cache 14. If (at block 458) there are any requested tracks not found in the first 14 and second 18 caches, then the cache manager 24 determines (at block 460) any of the requested tracks in the storage 10, from the second cache track index 70, not in the first 14 and the second 18 caches. The cache manager 24 then promotes (at block 462) any of the

determined tracks in the second cache 18 and the storage 10 to the first cache 14. The cache manager 24 uses (at block 464) the first cache track index 50 to retrieve the requested tracks from the first cache 14 to return to the read request. The entries for the retrieved tracks are moved (at block 466) to the MRU end of the LRU list 54, 56, 58 including entries for the retrieved tracks.

[0054] With the operations of FIG. 13, the cache manager 24 retrieves requested tracks from a highest level cache 14, then second cache 18 first before going to the storage 10, because the caches 14 and 18 would have the most recent modified version of a requested track. The most recent version is first found in the first cache 14, then the second cache 18 if not in the first cache 14 and then the storage 10 if not in either cache 14, 18.

[0055] Described embodiments provide techniques to group tracks in a first cache in strides defined according to a RAID configuration for the second cache, so that tracks in the first cache can be grouped in strides to a second cache. The tracks cached in the second cache may then be grouped into strides, defined according to a RAID configuration for the storage, and then written to the storage system.

[0056] Described embodiments provide techniques for promoting tracks from a first cache in strides so that the tracks may be written as full stride writes to strides in the second cache to improve the efficiency of cache promotion operations. The described embodiments allow full stride writes to be used to promote demoted tracks in the first cache to the second cache in order to conserve resources by promoting an entire stride to the second cache as a single I/O operation.

[0057] Further, while tracks are being promoted from the first cache 14 to the second cache 18 as strides, tracks are demoted from the second cache 18 on a track-by-track basis according to a cache demotion algorithm, such as an LRU algorithm.

[0058] The described operations may be implemented as a method, apparatus or computer program product using standard programming and/or engineering techniques to produce software, firmware, hardware, or any combination thereof. Accordingly, aspects of the

embodiments may take the form of an entirely hardware embodiment, an entirely software embodiment (including firmware, resident software, micro-code, etc.) or an embodiment combining software and hardware aspects that may all generally be referred to herein as a "circuit," "module" or "system." Furthermore, aspects of the embodiments may take the form of a computer program product embodied in one or more computer readable medium(s) having computer readable program code embodied thereon.

[0059] Any combination of one or more computer readable medium(s) may be utilized. The computer readable medium may be a computer readable signal medium or a computer readable storage medium. A computer readable storage medium may be, for example, but not limited to, an electronic, magnetic, optical, electromagnetic, infrared, or semiconductor system, apparatus, or device, or any suitable combination of the foregoing. More specific examples (a non-exhaustive list) of the computer readable storage medium would include the following: an electrical connection having one or more wires, a portable computer diskette, a hard disk, a random access memory (RAM), a read-only memory (ROM), an erasable programmable read-only memory (EPROM or Flash memory), an optical fiber, a portable compact disc read-only memory (CD-ROM), an optical storage device, a magnetic storage device, or any suitable combination of the foregoing. In the context of this document, a computer readable storage medium may be any tangible medium that can contain or store a program for use by or in connection with an instruction execution system, apparatus, or device.

[0060] A computer readable signal medium may include a propagated data signal with computer readable program code embodied therein, for example, in baseband or as part of a carrier wave. Such a propagated signal may take any of a variety of forms, including, but not limited to, electro-magnetic, optical, or any suitable combination thereof. A computer readable signal medium may be any computer readable medium that is not a computer readable storage medium and that can communicate, propagate, or transport a program for use by or in connection with an instruction execution system, apparatus, or device

[0061] Program code embodied on a computer readable medium may be transmitted using any appropriate medium, including but not limited to wireless, wireline, optical fiber cable, RF, etc., or any suitable combination of the foregoing.

[0062] Computer program code for carrying out operations for aspects of the present invention may be written in any combination of one or more programming languages, including an object oriented programming language such as Java, Smalltalk, C++ or the like and conventional procedural programming languages, such as the "C" programming language or similar programming languages. The program code may execute entirely on the user's computer, partly on the user's computer, as a stand-alone software package, partly on the user's computer and partly on a remote computer or entirely on the remote computer or server. In the latter scenario, the remote computer may be connected to the user's computer through any type of network, including a local area network (LAN) or a wide area network (WAN), or the connection may be made to an external computer (for example, through the Internet using an Internet Service Provider).

[0063] Aspects of the present invention are described above with reference to flowchart illustrations and/or block diagrams of methods, apparatus (systems) and computer program products according to embodiments of the invention. It will be understood that each block of the flowchart illustrations and/or block diagrams, and combinations of blocks in the flowchart illustrations and/or block diagrams, can be implemented by computer program instructions. These computer program instructions may be provided to a processor of a general purpose computer, special purpose computer, or other programmable data processing apparatus to produce a machine, such that the instructions, which execute via the processor of the computer or other programmable data processing apparatus, create means for implementing the functions/acts specified in the flowchart and/or block diagram block or blocks.

[0064] These computer program instructions may also be stored in a computer readable medium that can direct a computer, other programmable data processing apparatus, or other devices to function in a particular manner, such that the instructions stored in the computer

readable medium produce an article of manufacture including instructions which implement the function/act specified in the flowchart and/or block diagram block or blocks.

[0065] The computer program instructions may also be loaded onto a computer, other programmable data processing apparatus, or other devices to cause a series of operational steps to be performed on the computer, other programmable apparatus or other devices to produce a computer implemented process such that the instructions which execute on the computer or other programmable apparatus provide processes for implementing the functions/acts specified in the flowchart and/or block diagram block or blocks.

[0066] The terms "an embodiment", "embodiment", "embodiments", "the embodiment", "the embodiments", "one or more embodiments", "some embodiments", and "one embodiment" mean "one or more (but not all) embodiments of the present invention(s)" unless expressly specified otherwise.

[0067] The terms "including", "comprising", “having” and variations thereof mean "including but not limited to", unless expressly specified otherwise.

[0068] The enumerated listing of items does not imply that any or all of the items are mutually exclusive, unless expressly specified otherwise.

[0069] The terms "a", "an" and "the" mean "one or more", unless expressly specified otherwise.

[0070] Devices that are in communication with each other need not be in continuous communication with each other, unless expressly specified otherwise. In addition, devices that are in communication with each other may communicate directly or indirectly through one or more intermediaries.

[0071] A description of an embodiment with several components in communication with each other does not imply that all such components are required. On the contrary a variety of

optional components are described to illustrate the wide variety of possible embodiments of the present invention.

[0072] Further, although process steps, method steps, algorithms or the like may be described in a sequential order, such processes, methods and algorithms may be configured to work in alternate orders. In other words, any sequence or order of steps that may be described does not necessarily indicate a requirement that the steps be performed in that order. The steps of processes described herein may be performed in any order practical. Further, some steps may be performed simultaneously.

[0073] When a single device or article is described herein, it will be readily apparent that more than one device/article (whether or not they cooperate) may be used in place of a single device/article. Similarly, where more than one device or article is described herein (whether or not they cooperate), it will be readily apparent that a single device/article may be used in place of the more than one device or article or a different number of devices/articles may be used instead of the shown number of devices or programs. The functionality and/or the features of a device may be alternatively embodied by one or more other devices which are not explicitly described as having such functionality/features. Thus, other embodiments of the present invention need not include the device itself.

[0074] The illustrated operations of the figures show certain events occurring in a certain order. In alternative embodiments, certain operations may be performed in a different order, modified or removed. Moreover, steps may be added to the above described logic and still conform to the described embodiments. Further, operations described herein may occur sequentially or certain operations may be processed in parallel. Yet further, operations may be performed by a single processing unit or by distributed processing units.

[0075] The foregoing description of various embodiments of the invention has been presented for the purposes of illustration and description. It is not intended to be exhaustive or to limit the invention to the precise form disclosed. Many modifications and variations are possible in light of the above teaching. It is intended that the scope of the invention be limited not by this detailed description, but rather by the claims appended hereto. The above

specification, examples and data provide a complete description of the manufacture and use of the composition of the invention. Since many embodiments of the invention can be made without departing from the spirit and scope of the invention, the invention resides in the claims herein after appended.

CLAIMS

1. A computer program product for managing data in a cache system comprising a first cache, a second cache, and a storage system, the computer program product comprising a computer readable storage medium having computer readable program code embodied therein that executes to perform operations, the operations comprising:

determining tracks stored in the storage system to demote from the first cache;

forming a first stride including the determined tracks to demote;

determining a second stride in the second cache in which to include the tracks in the first stride;

adding the tracks from the first stride to the second stride in the second cache;

determining tracks in strides in the second cache to demote from the second cache; and

demoting the determined tracks to demote from the second cache.

2. The computer program product of claim 1, wherein the first cache is a faster access device than the second cache and wherein the second cache is a faster access device than the storage system.

3. The computer program product of claim 1, wherein the first cache comprises a Dynamic Random Access Memory (RAM), the second cache comprises a plurality of flash devices, and the storage system is comprised of a plurality of slower access devices than the flash devices.

4. The computer program product of claim 1, wherein the determined tracks to demote from the second cache are from different strides in the second cache.

5. The computer program product of claim 1, wherein the operations further comprise:

indicating the determined tracks to demote from the second cache as invalid, wherein strides in the second cache include valid and invalid tracks.

6. The computer program product of claim 1, wherein the operations further comprise:

receiving a write to a track in the first cache;

determining whether the track receiving the write is included in the second cache; and

invalidating the track in the second cache updated in the first cache in response to determining that the track written to in the first cache is included in the second cache.

7. The computer program product of claim 1, wherein the operations further comprise:

determining whether to consolidate strides in the second cache;

in response to determining to consolidate strides, performing:

determining one available stride having no tracks;

determining at least two strides having both valid and invalid tracks;

combining the valid tracks from at least two strides into the determined available stride, wherein the at least two strides are available to store tracks from strides demoted from the first cache.

8. The computer program product of claim 7, wherein the determination of whether to consolidate strides is performed in response to determining that there is only one available stride having all free tracks after writing the tracks in the stride demoted from the first cache to one available stride in the second cache.

9. The computer program product of claim 1, wherein forming the first stride of tracks comprising forming a stride for a Redundant Array of Independent Disk (RAID) configuration based on a RAID configuration defined for the second cache as having n devices including m devices for storing tracks of data and at least one parity device to store parity data calculated from the tracks of data for the m devices.

10. The computer program product of claim 9, wherein the first cache comprises a dynamic random access memory (DRAM), and wherein the second cache comprises n solid state storage devices, wherein the first stride of tracks from the first cache are striped across the n solid state storage devices to form the second stride in the second cache.

11. The computer program product of claim 1, wherein the operations further comprise:

using a first least recently used (LRU) list for tracks in the first cache, wherein the tracks to demote are determined from the first LRU list; and

using a second LRU list for tracks in the second cache to determine tracks to demote from the second cache.

12. The computer program product of claim 11, wherein the first cache stores tracks comprising modified or unmodified data and wherein tracks formed in strides in the first cache to promote to the second cache comprise unmodified data, wherein the operations further comprise:

destaging a modified track from the first cache to the storage; and

indicating the destaged modified track as an unmodified track, wherein the destaged modified track indicated as the unmodified track is eligible to be included in the strides promoted to the second cache.

13. A system in communication with a storage system, comprising:

a processor;

a first cache accessible to the processor;

a second cache accessible to the processor'

a computer readable storage medium having computer readable program code embodied therein executed by the processor to perform operations, the operations comprising:

determining tracks stored in the storage system to demote from the first cache;

forming a first stride including the determined tracks to demote;

determining a second stride in the second cache in which to include the tracks in the first stride;

adding the tracks from the first stride to the second stride in the second cache;

determining tracks in strides in the second cache to demote from the second cache; and

demoting the determined tracks to demote from the second cache.

14. The system of claim 13, wherein the first cache is a faster access device than the second cache and wherein the second cache is a faster access device than the storage system.

15. The system of claim 13, wherein the determined tracks to demote from the second cache are from different strides in the second cache.

16. The system of claim 13, wherein the operations further comprise:

receiving a write to a track in the first cache;

determining whether the track receiving the write is included in the second cache; and

invalidating the track in the second cache updated in the first cache in response to determining that the track written to in the first cache is included in the second cache.

17. The system of claim 13, wherein the operations further comprise:

determining whether to consolidate strides in the second cache;

in response to determining to consolidate strides, performing:

determining one available stride having no tracks;

determining at least two strides having both valid and invalid tracks;

combining the valid tracks from at least two strides into the determined available stride, wherein the at least two strides are available to store tracks from strides demoted from the first cache.

18. The system of claim 13, wherein forming the first stride of tracks comprising forming a stride for a Redundant Array of Independent Disk (RAID) configuration based on a RAID configuration defined for the second cache as having n devices including m devices for storing tracks of data and at least one parity device to store parity data calculated from the tracks of data for the m devices.

19. The system of claim 13, wherein the operations further comprise:

using a first least recently used (LRU) list for tracks in the first cache, wherein the tracks to demote are determined from the first LRU list; and

using a second LRU list for tracks in the second cache to determine tracks to demote from the second cache.

20. The system of claim 19, wherein the first cache stores tracks comprising modified or unmodified data and wherein tracks formed in strides in the first cache to promote to the second cache comprise unmodified data, wherein the operations further comprise:

destaging a modified track from the first cache to the storage; and

indicating the destaged modified track as an unmodified track, wherein the destaged modified track indicated as the unmodified track is eligible to be included in the strides promoted to the second cache.

21. A method for managing data in a cache system, comprising:

determining tracks stored in a storage system to demote from a first cache;

forming a first stride including the determined tracks to demote;

determining a second stride in a second cache in which to include the tracks in the first stride;

adding the tracks from the first stride to the second stride in the second cache;

determining tracks in strides in the second cache to demote from the second cache; and

demoting the determined tracks to demote from the second cache.

22. The method of claim 21, wherein the first cache is a faster access device than the second cache and wherein the second cache is a faster access device than the storage system.

23. The method of claim 21, wherein the determined tracks to demote from the second cache are from different strides in the second cache.

24. The method of claim 21, further comprising:

receiving a write to a track in the first cache;

determining whether the track receiving the write is included in the second cache; and

invalidating the track in the second cache updated in the first cache in response to determining that the track written to in the first cache is included in the second cache.

25. The method of claim 21, further comprising:

determining whether to consolidate strides in the second cache;

in response to determining to consolidate strides, performing:

determining one available stride having no tracks;

determining at least two strides having both valid and invalid tracks;

combining the valid tracks from at least two strides into the determined available stride, wherein the at least two strides are available to store tracks from strides demoted from the first cache.

26. The method of claim 21, wherein forming the first stride of tracks comprising forming a stride for a Redundant Array of Independent Disk (RAID) configuration based on a RAID configuration defined for the second cache as having n devices including m devices for storing tracks of data and at least one parity device to store parity data calculated from the tracks of data for the m devices.

27. The method of claim 21, further comprising:

using a first least recently used (LRU) list for tracks in the first cache, wherein the tracks to demote are determined from the first LRU list; and

using a second LRU list for tracks in the second cache to determine tracks to demote from the second cache.

28. The method of claim 27, wherein the first cache stores tracks comprising modified or unmodified data and wherein tracks formed in strides in the first cache to promote to the second cache comprise unmodified data, further comprising:

destaging a modified track from the first cache to the storage; and

indicating the destaged modified track as an unmodified track, wherein the destaged modified track indicated as the unmodified track is eligible to be included in the strides promoted to the second cache.

29. The system of claim 13, wherein the first cache comprises a Dynamic Random Access Memory (DRAM), the second cache comprises a plurality of flash devices, and the storage system is comprised of a plurality of slower access devices than the flash devices.

30. The system of claim 13, wherein the operations further comprise:

indicating the determined tracks to demote from the second cache is invalid, wherein strides in the second cache include valid and invalid tracks.

31. The system of claim 17, wherein the determination of whether to consolidate strides is performed in response to determining that there is only one available stride having all free tracks after writing the tracks in the stride demoted from the first cache to one available stride in the second cache.

32. The system of claim 18, wherein the first cache comprises a Dynamic Random Access Memory (DRAM), and wherein the second cache comprises n solid state storage devices, wherein the first stride of tracks from the first cache are striped across the n solid state storage devices to form the second stride in the second cache.

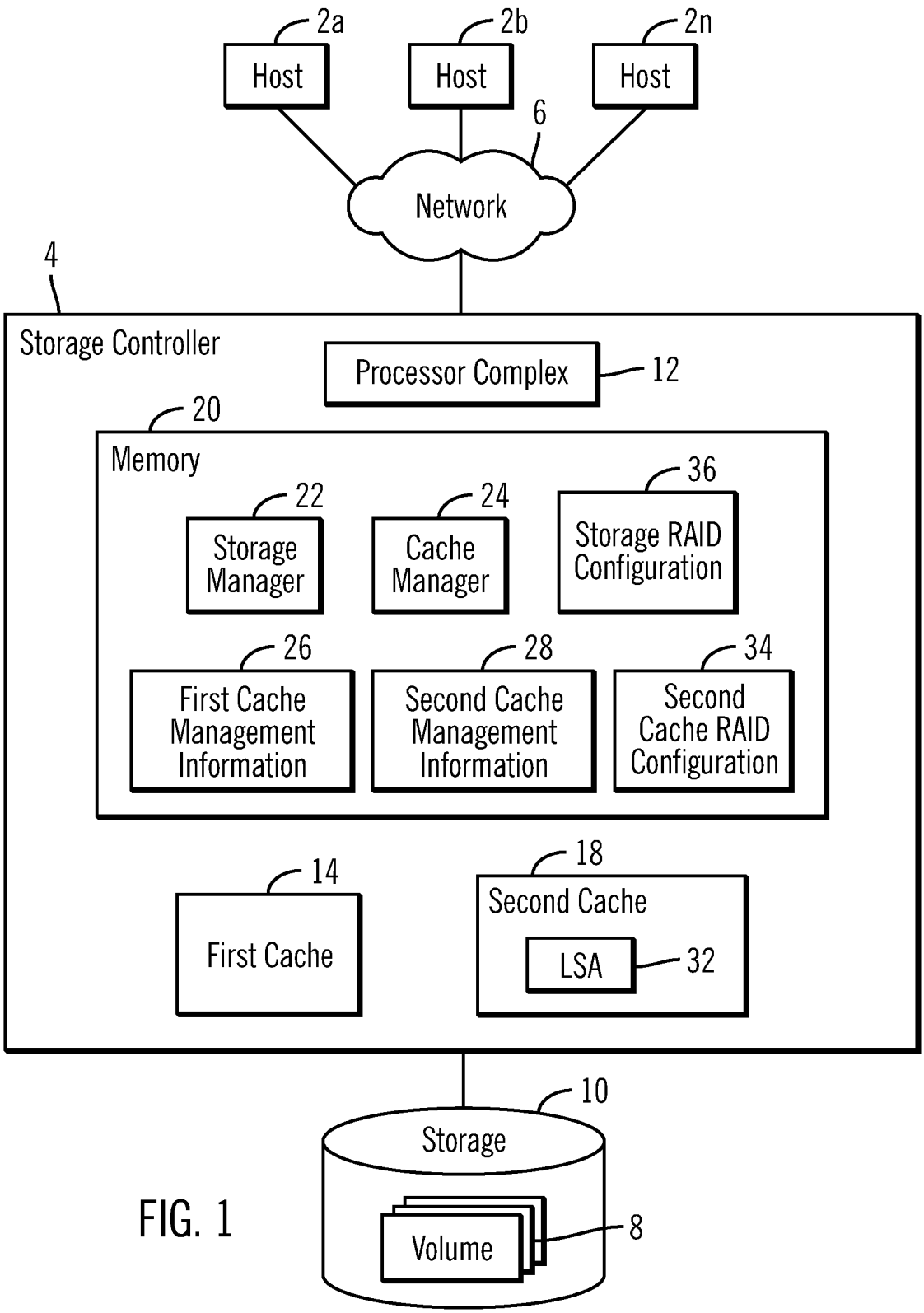

2a
Host
2b
Host
2n
Host
6
Network
4
Storage Controller
Processor Complex
12
20
Memory
22
Storage Manager
24
Cache Manager
36
Storage RAID Configuration
26
First Cache Management Information
28
Second Cache Management Information
34
Second Cache RAID Configuration
14
First Cache
18
Second Cache
LSA
32
10
Storage
Volume
8

FIG. 1

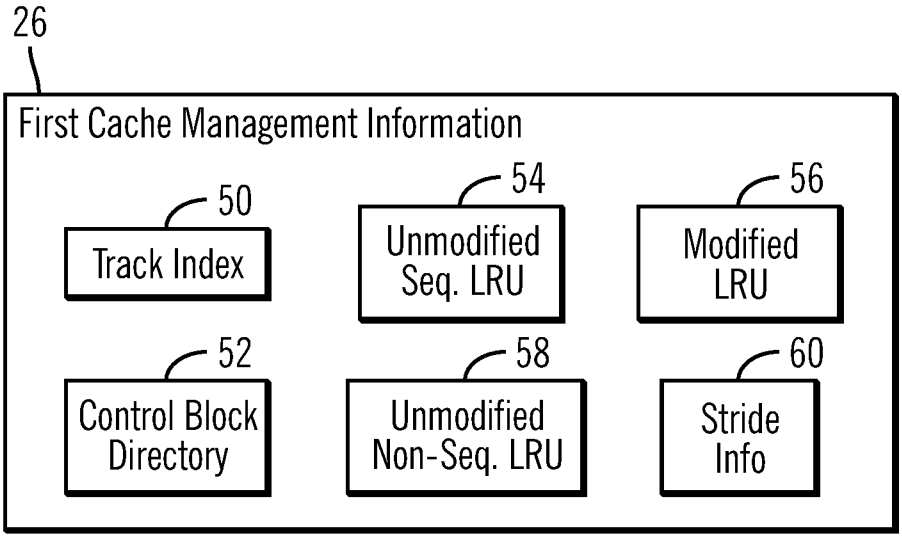

26
First Cache Management Information
50
Track Index
54
Unmodified Seq. LRU
56
Modified LRU
52
Control Block Directory
58
Unmodified Non-Seq. LRU
60
Stride Info

FIG. 2

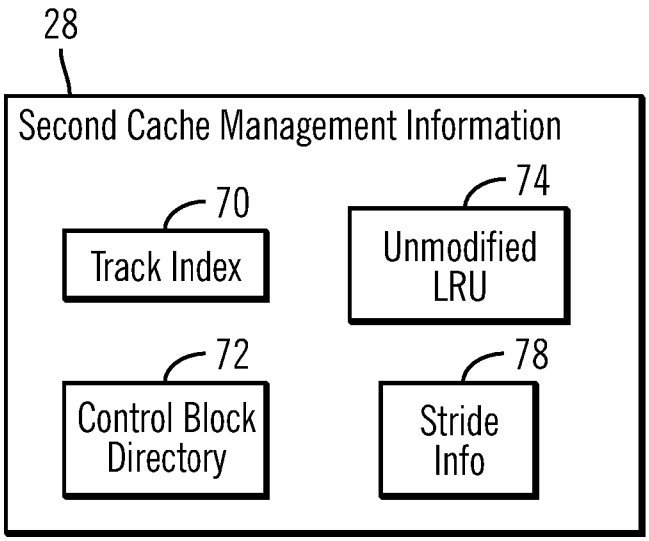
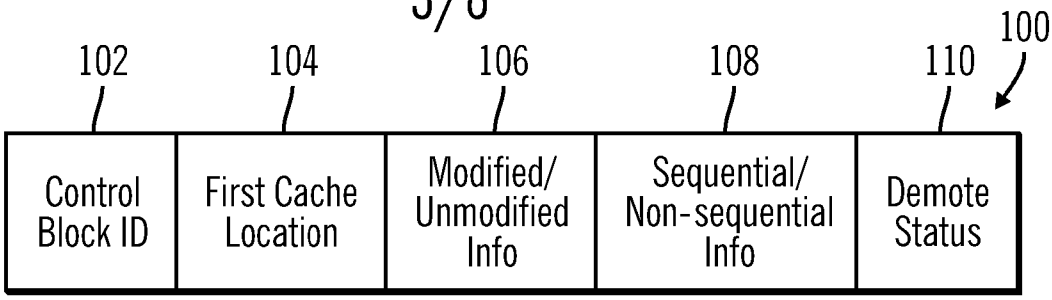

28
Second Cache Management Information
70
Track Index
74
Unmodified LRU
72
Control Block Directory
78
Stride Info

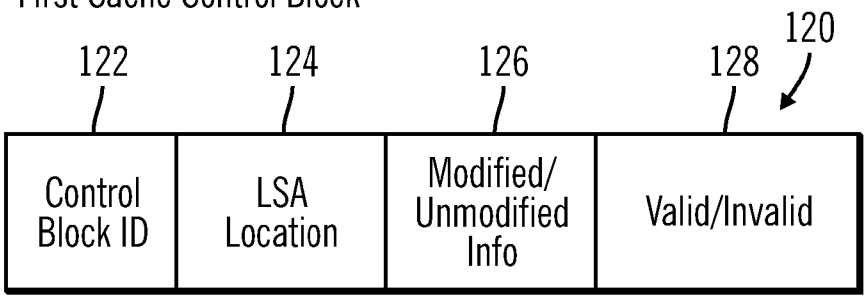
FIG. 3

FIG. 4



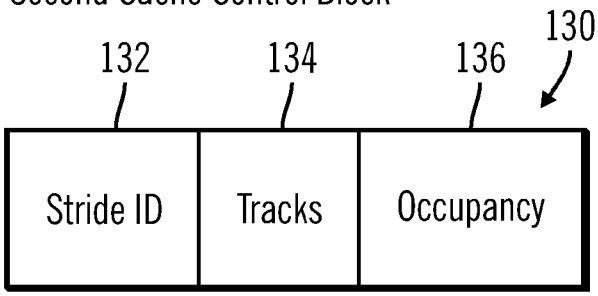
First Cache Control Block

FIG. 5



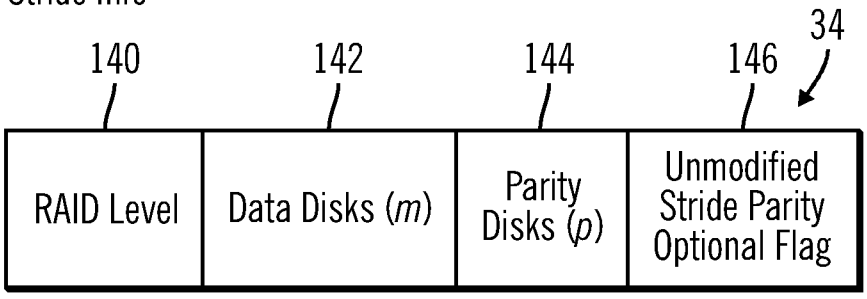
Second Cache Control Block

FIG. 6



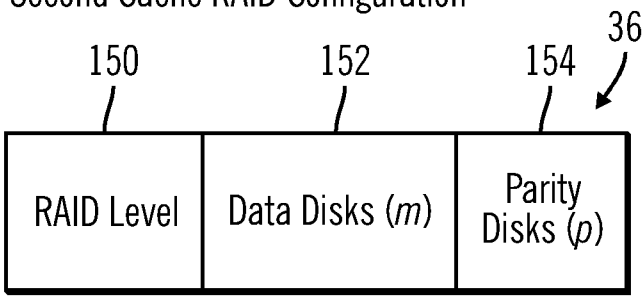
Stride Info

FIG. 7



Second Cache RAID Configuration

FIG. 8



Storage System RAID Configuration

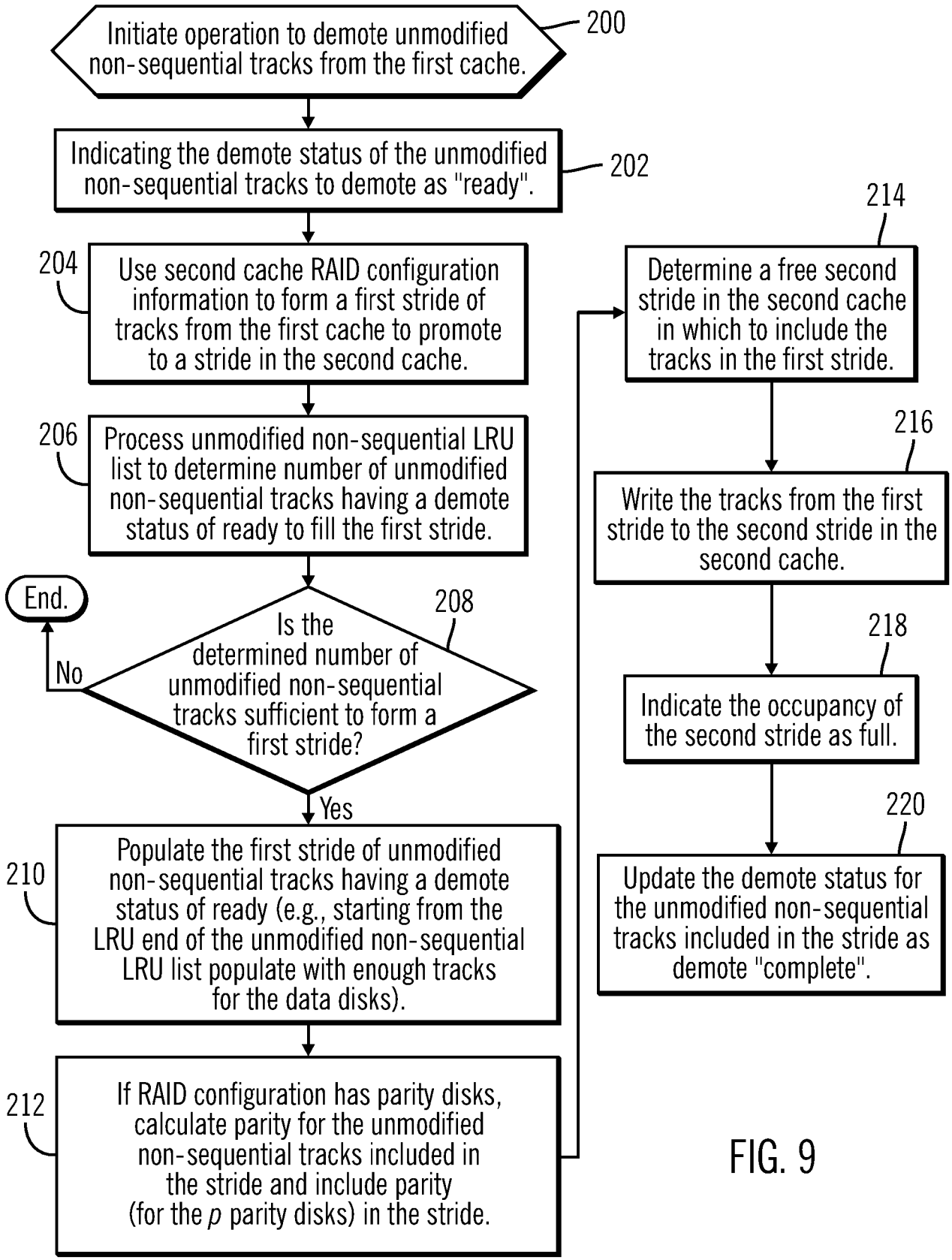

Initiate operation to demote unmodified non-sequential tracks from the first cache.
200
Indicating the demote status of the unmodified non-sequential tracks to demote as "ready".
202
204
Use second cache RAID configuration information to form a first stride of tracks from the first cache to promote to a stride in the second cache.
206
Process unmodified non-sequential LRU list to determine number of unmodified non-sequential tracks having a demote status of ready to fill the first stride.
208
Is the determined number of unmodified non-sequential tracks sufficient to form a first stride?
No
End.
Yes
210
Populate the first stride of unmodified non-sequential tracks having a demote status of ready (e.g., starting from the LRU end of the unmodified non-sequential LRU list populate with enough tracks for the data disks).
212
If RAID configuration has parity disks, calculate parity for the unmodified non-sequential tracks included in the stride and include parity (for the p parity disks) in the stride.
214
Determine a free second stride in the second cache in which to include the tracks in the first stride.
216
Write the tracks from the first stride to the second stride in the second cache.
218
Indicate the occupancy of the second stride as full.
220
Update the demote status for the unmodified non-sequential tracks included in the stride as demote "complete".

FIG. 9

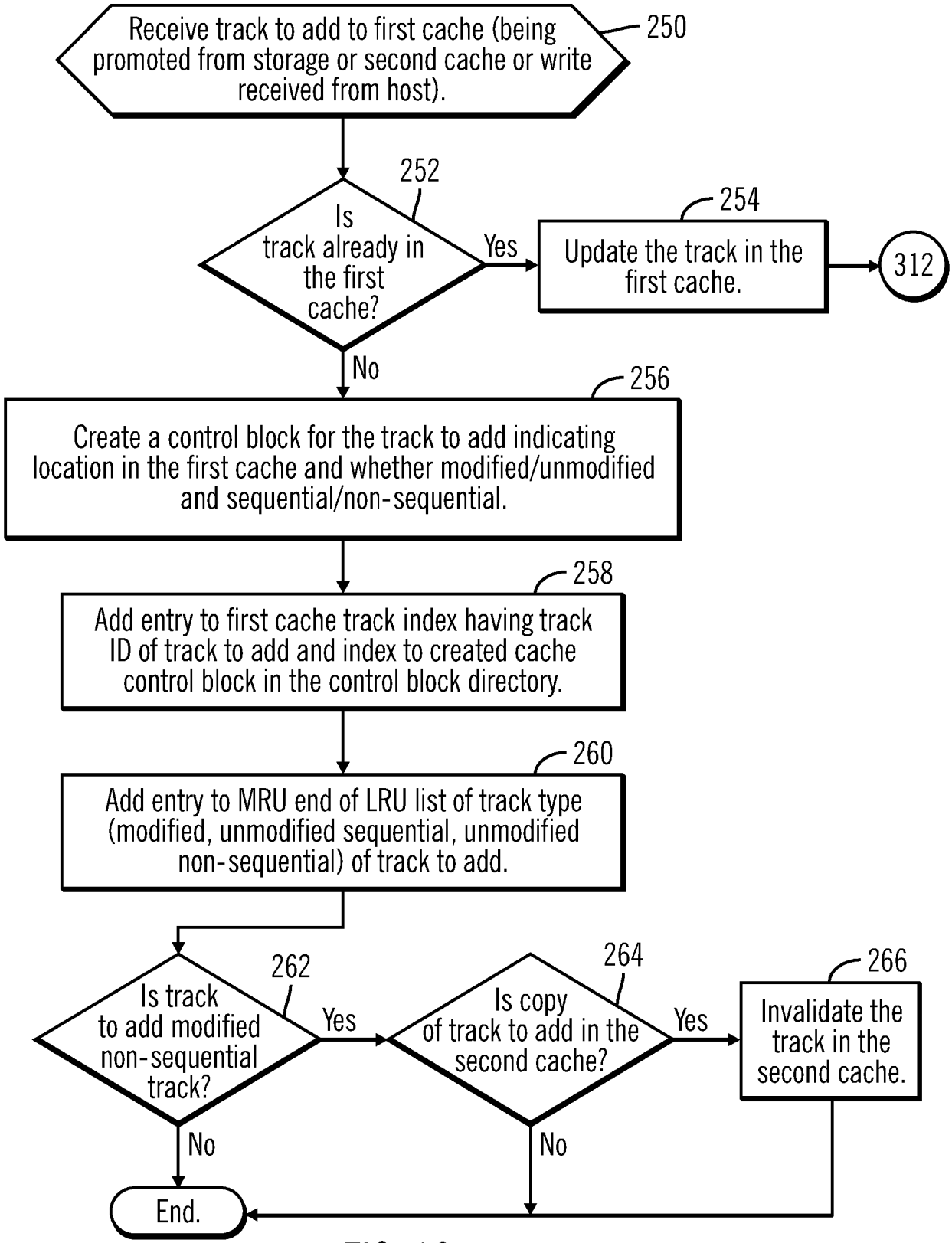

Receive track to add to first cache (being promoted from storage or second cache or write received from host).
250
252
Is track already in the first cache?
Yes
254
Update the track in the first cache.
312
No
256
Create a control block for the track to add indicating location in the first cache and whether modified/unmodified and sequential/non-sequential.
258
Add entry to first cache track index having track ID of track to add and index to created cache control block in the control block directory.
260
Add entry to MRU end of LRU list of track type (modified, unmodified sequential, unmodified non-sequential) of track to add.
262
Is track to add modified non-sequential track?
Yes
264
Is copy of track to add in the second cache?
Yes
266
Invalidate the track in the second cache.
No
No
End.

FIG. 10

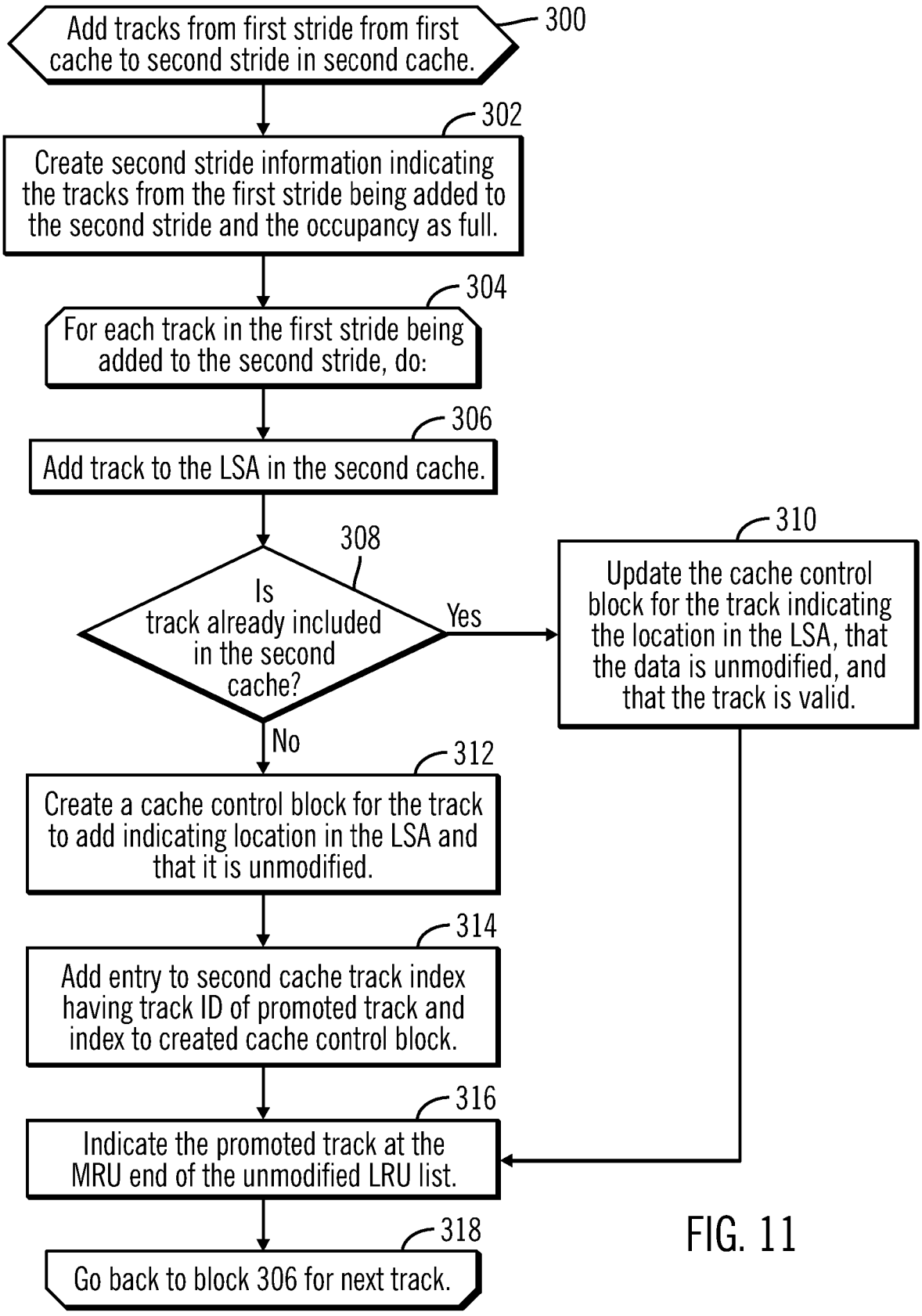

Add tracks from first stride from first cache to second stride in second cache.
300
Create second stride information indicating the tracks from the first stride being added to the second stride and the occupancy as full.
302
For each track in the first stride being added to the second stride, do:
304
Add track to the LSA in the second cache.
306
Is track already included in the second cache?
308
Yes
No
Update the cache control block for the track indicating the location in the LSA, that the data is unmodified, and that the track is valid.
310
Create a cache control block for the track to add indicating location in the LSA and that it is unmodified.
312
Add entry to second cache track index having track ID of promoted track and index to created cache control block.
314
Indicate the promoted track at the MRU end of the unmodified LRU list.
316
Go back to block 306 for next track.
318

FIG. 11

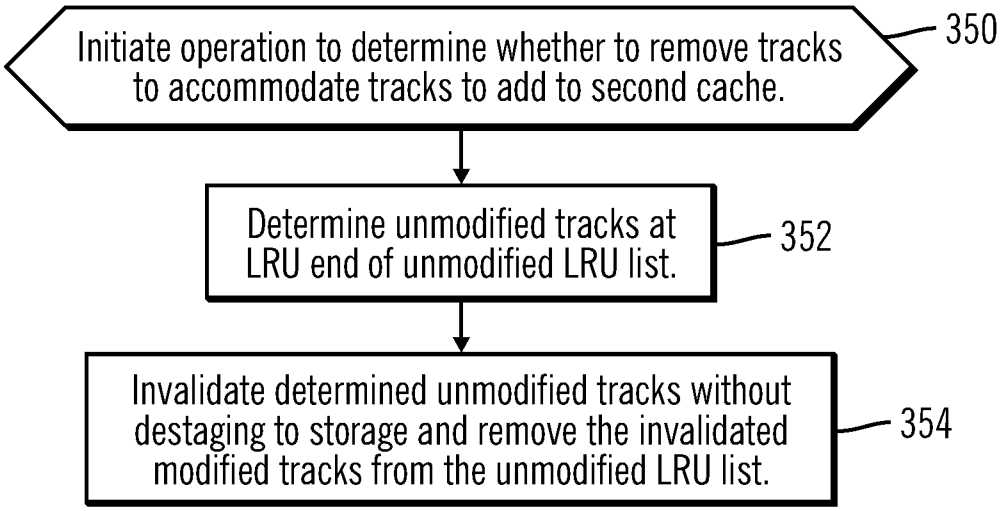

Initiate operation to determine whether to remove tracks to accommodate tracks to add to second cache.
350
Determine unmodified tracks at LRU end of unmodified LRU list.
352
Invalidate determined unmodified tracks without destaging to storage and remove the invalidated modified tracks from the unmodified LRU list.
354

FIG. 12

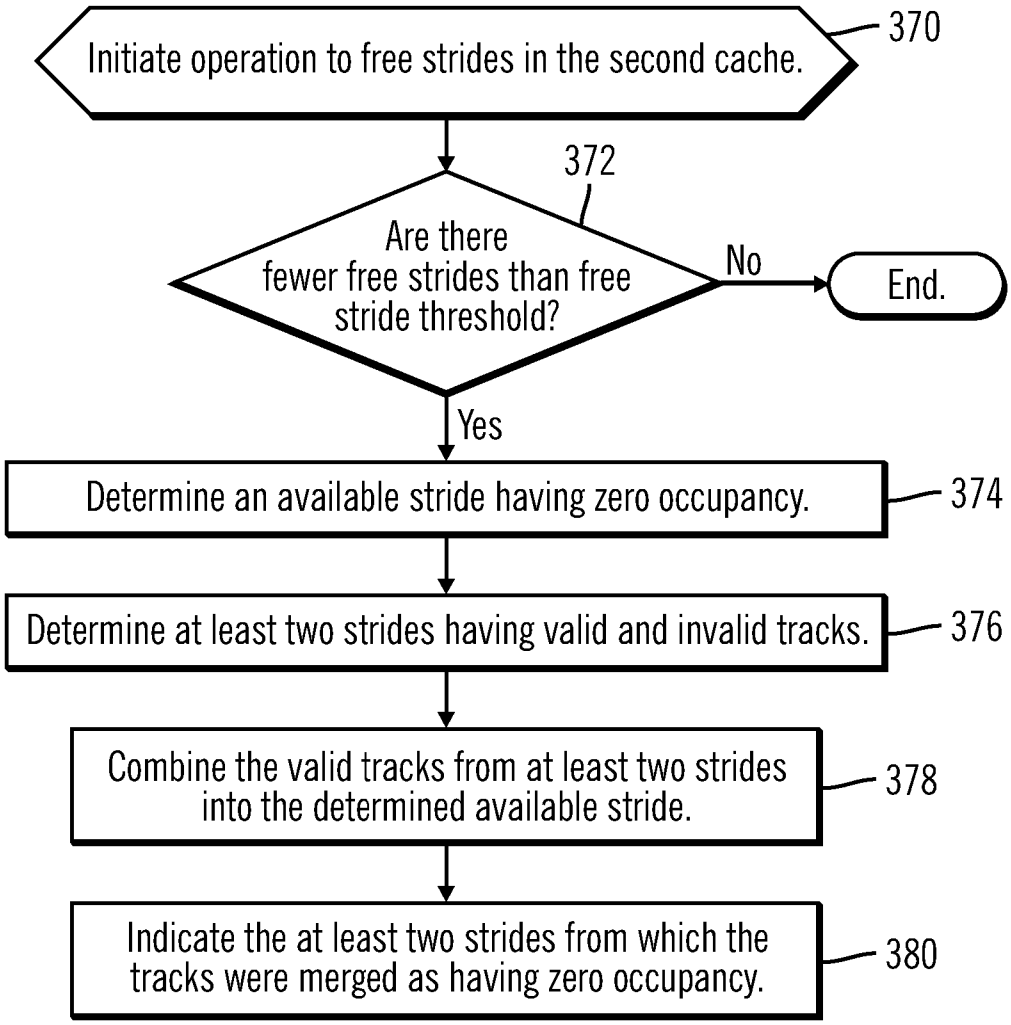

Initiate operation to free strides in the second cache.
370
372
Are there fewer free strides than free stride threshold?
No
End.
Yes
Determine an available stride having zero occupancy.
374
Determine at least two strides having valid and invalid tracks.
376
Combine the valid tracks from at least two strides into the determined available stride.
378
Indicate the at least two strides from which the tracks were merged as having zero occupancy.
380

FIG. 13

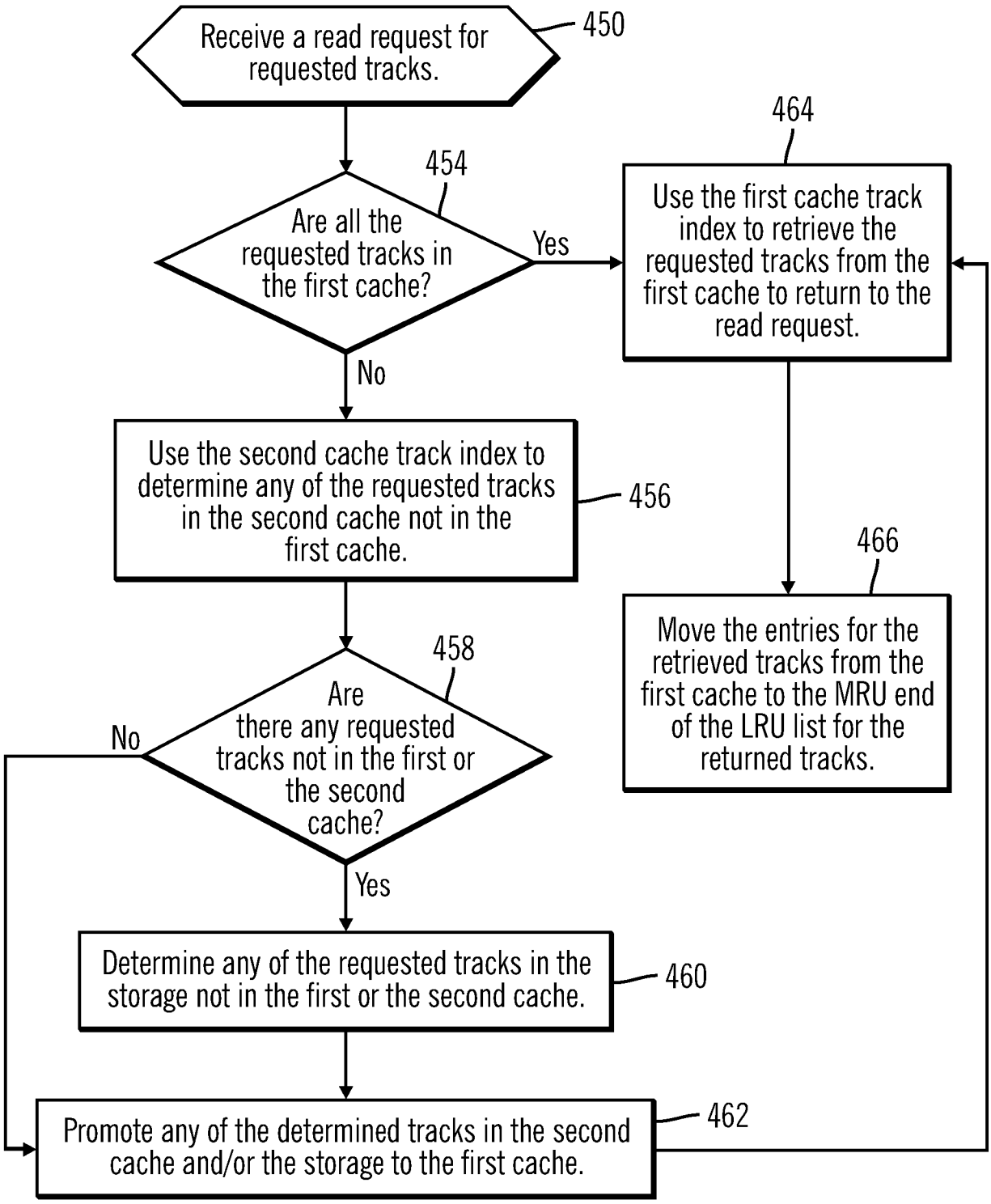

Receive a read request for requested tracks.
450
Are all the requested tracks in the first cache?
454
Yes
No
Use the first cache track index to retrieve the requested tracks from the first cache to return to the read request.
464
Use the second cache track index to determine any of the requested tracks in the second cache not in the first cache.
456
Move the entries for the retrieved tracks from the first cache to the MRU end of the LRU list for the returned tracks.
466
Are there any requested tracks not in the first or the second cache?
458
No
Yes
Determine any of the requested tracks in the storage not in the first or the second cache.
460
Promote any of the determined tracks in the second cache and/or the storage to the first cache.
462

FIG. 14

INTERNATIONAL SEARCH REPORT

International application No.
PCT/IB2012/057140

A. CLASSIFICATION OF SUBJECT MATTER

G06F 12/08 (2006.01) i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC: G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

CNPAT, CNKI, WPI, EPODOC, IEEE: cache, stride, demote, least w recently w used, LRU, track, destage

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	CN 1967507 A (IBM CORP.) 23 May 2007 (23.05.2007) page 4, line 18 to page 7, line 5, and figures 3 to 7	1-32
A	CN 1967495 A (IBM CORP.) 23 May 2007 (23.05.2007) the whole document	1-32
A	WO 2011042428 A1 (IBM CORP.) 14 April 2011 (14.04.2011) the whole document	1-32
A	US 2009216954 A1 (IBM CORP.) 27 August 2009 (27.08.2009) the whole document	1-32

☐ Further documents are listed in the continuation of Box C. ☒ See patent family annex.

* Special categories of cited documents:

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

"E" earlier application or patent 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 when the document is taken alone

"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

Date of the actual completion of the international search	Date of mailing of the international search report
26 March 2013 (26.03.2013)	**02 May 2013 (02.05.2013)**
Name and mailing address of the ISA/CN The State Intellectual Property Office, the P.R.China 6 Xitucheng Rd., Jimen Bridge, Haidian District, Beijing, China 100088 Facsimile No. 86-10-62019451	Authorized officer LIU,Xu Telephone No. (86-10)62413618

Form PCT/ISA /210 (second sheet) (July 2009)

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/IB2012/057140

Patent Documents referred in the Report	Publication Date	Patent Family	Publication Date
CN 1967507 A	23.05.2007	US 2007118695 A1	24.05.2007
		JP 2007141225 A	07.06.2007
CN 1967495 A	23.05.2007	US 2007118698 A1	24.05.2007
		JP 2007141224 A	07.06.2007
WO 2011042428 A1	14.04.2011	US 2011087837 A1	14.04.2011
		US 2012191904 A1	26.07.2012
US 2009216954 A1	27.08.2009	None	