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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
9 June 2011 (09.06.2011)

PCT

(10) International Publication Number
WO 2011/068614 A2

(51) International Patent Classification:
G06F 13/16 (2006.01) *G06F 9/44* (2006.01)

(21) International Application Number:
PCT/US2010/055022

(22) International Filing Date:
1 November 2010 (01.11.2010)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/256,487 30 October 2009 (30.10.2009) US

(72) Inventor; and
(71) Applicant : **KULAKOWSKI, Robert** [US/US]; P.O. Box 471, Rancho Santa Fe, CA 92067 (US).

(74) Common Representative: **KULAKOWSKI, Robert**; P.O. Box 471, Rancho Santa Fe, CA 92067 (US).

(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, 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, PE, PG, PH, PL, PT, RO, RS, RU, 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, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, 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, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *of inventorship (Rule 4.17(iv))*

Published:

— *without international search report and to be republished upon receipt of that report (Rule 48.2(g))*

WO 2011/068614 A2

(54) Title: CONTROLLER DEVICE COPROCESSOR ARCHITECTURE

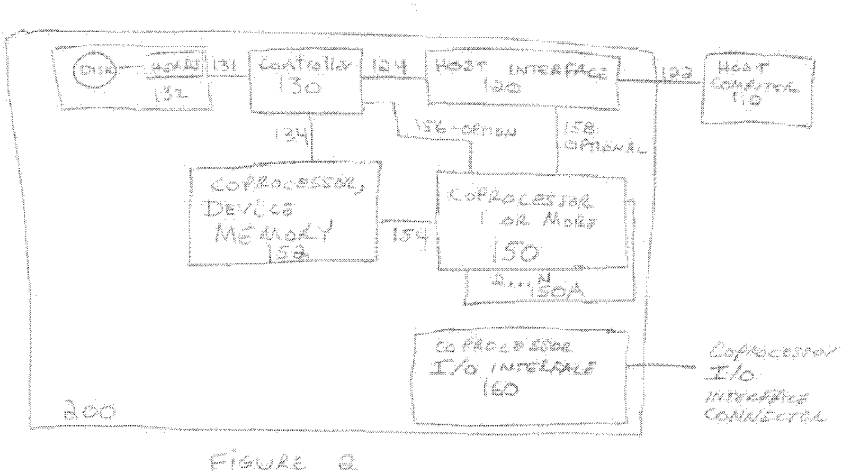

Figure 2

(57) Abstract: Device controllers are enhanced with application processors or coprocessors and additional input/output interfaces are added to the device. Coprocessors running application code allowing the device to serve as application servers for data stored on the device. Coprocessors include an application processing framework allowing access to device storage and optionally include additional memory busses and additional memory chips to reduce bus contention allowing coprocessors to achieve high throughput rates for inputting or outputting or processing device data. Coprocessors include software to access and process stored data without processing support from the host system interfacing to the device. Access to stored data is managed to prevent simultaneous access corruption when the data is being changed.

Background

This patent application is for an invention adding application processors also referred to as coprocessors to different types of controllers. An example device is referred to as an enhanced storage device such as a hard disk drive, or Solid State Disk drive or other type of computer or electronic device. In this patent application the term coprocessor or CP will be used to describe processing elements added to device controllers, and the CPs will runs application code different from normal device and device interface control code. Coprocessor can also be code running on a single controller shared with the software used for normal device control and interfacing. Normal device interface code for a SATA type hard disk drive is code that processes SATA data and commands connected to a host via a SATA interface. Coprocessor code is code that performs application level processing such as the RTSP protocol for a Video On Demand (VOD) server and the coprocessor code can run on the same physical CPU as the SATA processing code or on additional CPUs added to the device controller or device. In and RTSP (or other application) an processor external to the HDD may perform the subscriber authentication and establish the parameters for an RTSP session and then the RTSP protocol processing and content file stream is performed by the CP. This is one example of a distributed CP application wherein processing is distributed across a number of processing elements including one or more CPs. In this patent application the term host is used to refer to external processing element interfacing to HDD over host interface or another processing element interfacing to HDD over any other I/O interface such as an Ethernet port on HDD. Figures 1 through 20 are used to illustrate inventive concepts of the invention.

The examples in this patent application refer to HDD inventive elements and any of the inventive elements can also be applied to Flash Memory devices, Solid State Disk Drives and any type of storage device or controller or controller chip. Flash memory chip can be any type of solid-state memory such as NAND Flash, NOR Flash or other memory technology. The term storage assembly is used to describe the storage part

of a HDD or Flash memory array or Solid State memory device, or the memory storage portion of any storage device.

In this patent application the term element is used to describe any hardware or software in the system including but not limited to CPU, Coprocessor, application, task, interfaces, hosts, host processing, distributed processing hardware or software, etc.

In this patent application the term disk, hard disk, flash drive, SSD, hard disk drive, disk data, or HDD is used as exemplary language for any type of data storage or data processing device. Thus, when disk, hard disk, disk data, or HDD is used in examples the example also applies to any type of controller device or memory system including controllers for HDDs, Flash Memory drives, Flash Memory chips, Solid State Drives (SSDs) using solid state memory chips, tape drives, and other types of controllers used as part of systems or complete systems. This invention includes application processing support on the HDD unit or HDD controller eliminating the need to transfer certain data to an external processor for application level processing. The term application level processing is used to describe processing that is not normally performed on or in a HDD. Examples include but are not limited to portions or all of the processing for file server processing, video server processing, network file server, network file store, Storage Area Network (SAN) processing, broadcast ad splicing, targeted ad delivery, video personal video recorder processing, high-reliability HDD array processing, high speed disk to disk back up or mirroring, HDD raid processing, and other processing that is normally performed by a host 110 computer interfacing to a HDD via a host to HDD interface connection 122 and HDD host interface 120. It is envisioned that applications can be split between the HDD and host or other computers. For example, a Video On Demand (VOD)/RTSP/RTP application can have the subscriber access control processing performed on a computer external to the HDD and then the video file streaming processing controlled by an application processor (coprocessor) on the HDD, or both can run on HDD. In many examples in this patent application the term CPU will be used to describe a processor that is normally found in current day controller chips.

One application that demonstrated the improvements provided by this invention includes streaming video. In streaming video application HDDs are used only for storage of video data. Stored video data on the HDD is read by an application host processor running on a computer external to the HDD (VOD server for example) and then the application host processor formats the video data and then outputs the

formatted video data to a client device such as a television TV, personal computer PC, mobile phone, game player, etc. In a VOD server, video data stored on the HDD is served to a subscriber. Optimizing VOD server performance is typically performance limited by the performance of the host interface to the HDD and the processing performed by the Host of receiving data, formatting the data, and then outputting the data.

This invention adds application processing into the HDD itself eliminating the extra data processing and extra host processing required external to the HDD when compared to running the application software on the HDD itself.

In one example, a VOD server application is run on coprocessors and common content IDs are used for content delivery over managed networks such as IPTV networks or cable networks, and over the Internet (or over-the-top OTT) delivery of the same content. Content IDs for managed network delivery and Internet delivery may be different and an association of the different ID to one common content file is performed.

In another example, advertising splicing or targeted ad processing is performed by coprocessors.

Detailed description:

Figure 1 shows a current day HDD. In figure 1 storage device controller (controller 130) is the controller normally associated with device 100. When device 100 is a HDD controller 130 is a HDD controller, for a solid state memory device 100 controller 130 is a solid state memory device controller, and controller 130 is an appropriate controller for other types of devices 100. Device controller memory is memory used by controller 130. A host computer interface is shown as Host Interface 120 in Figure 1 and interfaces to a host computer.

Figure 2 shows a HDD enhanced with one or more application processors or coprocessors 150, 150A, 150B…150N, along with application memory or coprocessor memory, and additional optional interfaces 160. Each of these elements will be explained in detail below. Coprocessor memory 152 optionally contains all the different memory elements for the enhanced device described herein and is shared between application processor and storage device and optionally includes storage device memory buffer. Storage device memory buffer is a memory buffer used to read and write data to the storage assembly and in an HDD example is a 32 Megabyte buffer used to buffer data being written to the head assembly or read from the head

assembly. In this patent application the term element is used to describe any single item such as controller 130 or coprocessor 150 or other interface 160 or any combination of any item described in this patent application.

In Figure 2 various elements of the invention are shown as separate blocks, however this invention covers the combining of one or more of the components into a single chip. For example, controller 130 is usually combined with host interface 120 and one or more coprocessors 150 can be included in the same chip or housed externally to the single chip. Various memory elements (device controller memory, application program and data memory, buffer memory, etc.) can be contained in a single chip with a single memory bus interface or contained in one or more chips with one or more interface busses. The various blocks can be added to HDDs as a plug-in processor board via a connector on the HDD. In fact, the elements shown in Figure 2 can be split across multiple boards or implemented as a plug-in board for HDds. For example, coprocessor memory 152 and coprocessors 150 can be housed on a separate board from the HDD controller board and plugs into HDD controller board via a connector.

Disk assembly 132 includes disk heads, head positioning electromagnetic circuitry, and magnetic disk platters found in a HDD unit and typically interfaced to a HDD controller 130 and interconnected via connectors and interface signals 131. Disk data will refer generically to stored data and for HDDs refers to the digitally encoded data magnetically stored on a rotating platter or platters with magnetic surfaces, but also refers to data stored in solid state memory chips such as Flash memory (serial or parallel access), NAND Flash, NOR Flash, DRAM, SDRAM, static RAM memory, and other types of memory.

Host interface 120 is any form of interface that connects the HDD to host processor 110 (e.g. external computer). Examples of current day host interfaces include but are not limited to SATA, SCSI, IDE, ATA, SAS, etc. Host processor will read or write what is being called application data across interface 122 to host interface 120 in HDD. Application data is typically processed by controller 130 to add file system and disk level data such as sector numbers, sector data error correcting data, sector header data, etc., and this data will be referred to as raw HDD added data. So when data is written to the disk assembly 132 a combination of raw HDD data and application data is actually written to disk.

Hard disk data in disk assembly 132 is typically partitioned into sectors of data wherein a read/write head in disk assembly 132 reads or writes data to magnetic

surface of disk. As mentioned earlier this invention adds access control to coprocessor memory or disk sectors. The same type of access control or access protection described for coprocessor memory can be applied to the physical sectors on the disk, meaning that when the host 110 writes data to the disk assembly 132 the controller 130 can indicate the owner or source of the data (host 110 in this example) and limit access to that physical data to only the host 110. Likewise, HDD data (the data stored on the disk via disk heads) generated by a coprocessor or locked to a coprocessor 150 can become inaccessible to host 110 and other coprocessors 150 in a multiple coprocessor 150 system. In this patent application any form of access control (protected memory or a protected disk sector bound to one host 110 or one coprocessor 150) can be applied to coprocessor memory access or disk sector data or both. There are many ways access control can be implemented and one examples in the table below will be used to illustrate the processing. When multiple coprocessors are included in HDD any form or CP, memory, HDD data, SSD data, flash memory, non-volatile memory isolation can be implemented to prevent one CP or element from corrupting another CP or element or HDD data or host data or other data or program or configuration data.

Controller 130 is a controller of any form that performs control of standard current day HDDs. Typically controller 130 will perform processing with host 110 via host interface 120 and host interface connection 122 and perform control processing for the disk head assembly 132. Controller 130 connects to disk head assembly 132 via connection 131. The controller 130 is usually in the form of a single application specific IC (ASIC) that includes interface signals to disk head assembly 132 and performs low-level disk bitstream related control including adding disk data error correcting, performing head positioning control, processing commands from host 110 via host interface 120, and coordinating the HDD internal processing required to provide HDD functionality. Coprocessor 150 performs application level processing in addition to any disk level processing coprocessor 150 may provide. In many cases coprocessor 150 will not be required to assist controller 130 in control disk drive 200 and will be used mainly or fully for application processing for applications running on HDD. It is also envisioned that the coprocessor can optionally directly access HDD data without going through host interface.

Controller 130 or other element(s) in enhance HDD 200 includes processing to arbitrate HDD disk assembly 132 access between the various elements accessing HDD. There can be a plurality of elements simultaneously access hard disk assembly

132 for reading and writing data including but not limited to host 110, any one or more of the coprocessors 150, controller 130, and other hosts (not shown) accessing HDD 200 via additional optional interfaces 160. Any access arbitration technique can be applied including priority based arbitration or round robin arbitration, etc. Additional other optional interfaces 160 include any form of interface to multiple hosts (any interface including but not limited to SCSI, SATA, iSCSI, IDE, Fiber Channel, etc.) or for Ethernet interfacing or USB interfacing, or any other type, etc. Because coprocessors are adding application processing to HDD a single host interface 122 may not provide sufficient application level interfacing support. So, enhanced storage device (e.g. HDD) may include one or more application processor interfaces of any type such as Ethernet interface for example or one or more USB interfaces, or other types of i/o interfaces. It is envisioned that application processor input/output interface can be of any one or more of one type or multiple types of interfaces including but not limited to a single instance or multiple instances of Ethernet, multiple USB, WiFi, Firewire, etc. in addition to one or more host interfaces. Application processor interface can be a combination of a plurality of interfaces such as two Ethernet and two or more USB, or Firewire or other type. Other interfaces 160 may be used to send and receive any type of data including application control or configuration data, host interface data (bridging optional interface 160 to host), HDD data, or controller data.

Coprocessors are application software processors with low-level access to stored data in device Coprocessors 150 (also called APP CPUs) run application level software such as an RTSP server and access HDD memory for application level data. Coprocessors include software to run applications and have coprocessor software to access HDD or flash memory data used by applications running on coprocessor. In the RTSP example, the coprocessor will access movie files on HDD using either host interface on HDD or directly accessing HDD disk/memory data with file system knowledge to allow coprocessor to access disk/memory data storing movie file. Coprocessors can also access flash chip controllers without going through host interface for flash memory systems as shown in Figure 7.

Coprocessors Hard Disk Drive

Current day disk drive platters can store 2 to 4 terabytes of data and because of this large capacity two or more disk drive read/write head assemblies with one or more coprocessors provide increased disk drive throughput without the need for even

higher host interface interconnect bus speed such as SATA, SCSI, etc. In enterprise applications where one disk drive is shared amongst a number of users, a plurality of read/write heads and host interfaces using one magnetic disk platter array increases total hard disk drive performance without having to increase HDD interface speeds. Current day HDD interfaces speeds are in the range of 6 gigabits per second, and with 4 read/write head assemblies and 4 host interfaces 24.

One or more Controller(s) 130 will receive disk access commands and interface data over host interface 120 to host 100 and will also interface to coprocessors 150 and optional other interfaces 160. Controller 130 will provide command and data interfaces of any type including but not limited to SCSI, ATA, or other type of command/data interface to read/write data from disk head 132. While the actual command/data interface structure between coprocessors 150 and controller 130 can use any interface definition the coprocessors and external interfaces 160 will be capable of reading/writing disk data going through host interface 120 or through interface implemented in coprocessor memory 152 (memory based interface) or via an optional coprocessor 150 to controller 130 interface or an optional interface 160 to controller or combinations of interfaces. While in the examples in some cases only a single element is shown such as controller 130 this invention anticipates a plurality of elements such as two controllers 130 each interfacing to one or more head assemblies 132.

Another example includes two controllers 130 each individually interfaces to one of two disk head assemblies (second one not shown) allowing multiple coprocessors and hosts to simultaneously access different parts of the disk using multiple read/write head assemblies. In this configuration the HDD 200 will have double the disk read/write performance when compared to a single head assembly drive, because two head assemblies containing double the number of head of a normal drive can independently read/write disk data onto the magnetic disks. In normal HDDs this does not make sense because the host interface limits read/write performance but with multiple heads assemblies operating independently double performance can be achieved by the HDD with disk input output occurring over a plurality of host interfaces and coprocessors with external interfaces such as Ethernet interfaces. For example, if a HDD with four read/write heads can achieve 1,600 megabits of read/write speed of disk data, having two head assemblies each with four read/write heads will achieve 3,200 megabits of read/write speed for disk data. In this example a plurality of head assemblies (e.g. two but there can be more) coordinated access to the

magnetic data on the HDD. Coordinated access means that portions of the HDD disk data structures (sectors for example) are managed to prevent multiple head assemblies from writing the same data corrupting the drive data, and preventing a head assembly from reading data that is being written.

It is also envisioned by this application that coprocessors 150 contained on a single HDD or contained on multiple HDDs will operate redundantly to improve the availability of a service running on the HDD or plurality of HDDs. Redundancy can be based on disk data redundancy (disk data redundant across multiple HDDs with techniques such as RAID or providing equivalent redundancy) or redundant applications running on multiple coprocessors 150, or combinations of both. It is also envisioned that redundancy can be performed without any host processor or split between one or more HDDs with or without host processing support. When HDDs with multiple independent read/write heads and head assemblies share a common set of magnetic platters any form of disk data allocation can be incorporated including the techniques described herein or other techniques. HDD magnetic data can be allocated in any manner on a on an exclusive or non-exclusive basis to one or more elements or tasks or applications. Other forms of partitioning such as having one r/w head assembly be the master and a second a read-only slave, or having r/w head assembly locking for write access to any portions of the disk data including file allocation tables, file access rights data, and other file management related data or disk file data.

Write data controller – with read slaves

In some applications such as file server applications one or more of the system elements can act as a write-master for certain parts of disk data and the other system elements will have read-only access or no access at all. Disk data access rights and management can be managed on a dynamic basis wherein a system manager provides access tokens or assigns access rights data to other system elements allowing the system element to access protected or restricted data.

In some applications system elements will be processing disk data in parallel to other system elements and any form of system element data coordination to maintain coherent data across multiple system elements can be incorporated. When multiple system elements (such as APP CPUs, CPUs, Coprocessors, host interfaces) can access disk data any form of messaging or signaling can be used with data coherency must be maintained across the various system elements.

Coprocessors in SSD/Flash Memory Controllers

Figure 6 provides an example of a current day Solid State Disk (SSD) controller. Typical current day SSD controllers only incorporate one internal bus (Bus 620) interfacing to a plurality of Flash controller engines in the SSD controller. In Figure 6 N flash chips are supported by the SDD controller. SDD controller in Figure 6 has a single host interface 630, optionally a USB interface 632, a CPU 634 for internal processing and internal CPU/RAM/ROM 636 and a DRAM controller 638 to interface to an external DRAM memory chip used as a disk data buffer.

Flash controllers 1 610 through N 618 interface to external Flash memory chips shown as Flash Chip 1 through Flash Chip N.

Figure 7 provides and example of a SDD or flash controller enhance with coprocessors (APP CPU 1 650, APP CPU2 2 654, two Ethernet controllers (652, 656), and a second DRAM controller (DRAM Controller 2 658) along with the addition of two internal busses (BUS B 622 and BUS N 624) for a total of three internal busses. Figure 7 shows one example of an enhanced SDD controller and different combinations of one or more added CPUs, one or more internal busses, one or more Ethernet controllers, and one or more DRAM controllers are envisioned by this patent application.

In Figure 7 APP CPU1 is associated with Ethernet Controller 1 and a DRAM controller and interfaces to the Flash chips using one or more the internet busses (Bus A, Bus B, or Bus N). APP CPU1 runs code in parallel to APP CPU2 and the provided example also allows the host interface to operate concurrently with APP CPU1 and APP CPU2. Any form of Flash Memory chip partitioning can be incorporated including have one or more Flash Memory chips associated with the Host via Host Interface, or with an APP CPU, or combinations wherein one or more of the flash memory chips are accessible by any of the CPU/Host elements (host via host interface and App CPU1 and App CPU2). Other combinations are also anticipated where a Flash Memory chip can be dedicated to any one of the CPU/Host elements, or combinations can be incorporated where some of the Flash memory chips are associated with one or more of the CPU/Host elements and others are dedicated to only one of the CPU/Host elements, or combinations where the host interface can access all Flash Memory chips and APP CPUs are associated with specific memory chips. A bus to memory chip and APP CPU circuit design to maximize the total parallel throughput of one or more APP CPUs (or coprocessors) to memory chip data

is preferred. Ideally, the memory will be partitioned so that each APP CPU can access memory chip data concurrently at full speed with minimum bus contention between the APP CPUs and the host interface.

Disk or flash memory data is optionally allocated to any Coprocessor of Host Interface on a static or dynamic basis. Disk or flash memory management data includes control data to associate disk or flash memory to one or more host interfaces or one or more coprocessors on a shared or exclusive basis. Disk or flash memory allocation to host or coprocessors can be partitioned statically during initial configuration of the disk or flash memory or dynamically while the system is running. File storage across multiple controllers or multiple chips or multiple HDDs. For high data throughput and using a flash controller example, a plurality of Flash Controllers 700 are incorporated in a system with content split across the plurality of Flash Controllers 700. It is envisioned that a single movie file can be stored using two or more Flash Controllers. Figure 7A provides an example of a flash memory based system incorporating two enhanced flash controllers 700 and 702 and sixteen flash memory chips, four APP CPUs and four Ethernet controller interfaces (652, 656, 752, 756). To improve throughput a single content file (movie for example) is stored using all 16 flash chips spread across two flash controller chips. Flash controller 1 700 stores one part of the movie and Flash controller 2 702 stores the other part. Splitting a piece of content across two or more flash controllers 700, 702 allows for delivering the same piece of content to more uses with less bus and memory chip access content than when only one flash controller is used. When a plurality of flash controllers are used software coordinates the read and write accessing of data across multiple flash memory chips and across multiple flash controllers (e.g. 700 and 702). This coordination software can run on any processing element including APP CPUs, CPUs, or a processor external to flash controllers. Ethernet output from the plurality of flash controllers will contain proper IP and/or MAC addressing to appear as if the stream is originating from a single Ethernet source. An Ethernet switch (not shown) can also be incorporated to make the output stream appear as if it is from a single source. The output timing of the file when output from one or more controllers can optionally be time paced so that a predetermined output bandwidth can be provided. While the above examples uses outputting data as and example, the same techniques can be provided when writing data to the flash controller or a plurality of flash controllers wherein the plurality of flash controllers will appear to the outside world as a single device storing a single file and coordination software will partition the single file

across multiple flash chips and when incorporated multiple flash controllers. It is also envisioned in this patent application that elements in any flash controller (example flash controller 1 700) will coordinate read/write/input/output processing with elements in the same flash controller (APP CPU 1 or APP CPU 2), or in other flash controllers (e.g. flash controller2 702), and that flash memory related storage information such as file to storage mapping across multiple flash chips and multiple flash controllers (e.g. 700, 702) will be replicated, cached, or coordinated between the various elements (CPUs, CoProcessors, Flash Controllers, Flash Memory Chips, Interfaces, etc.) in one or more Flash Controllers.

Jitter free output switching between coprocessors

Figure 7A provides an example of a flash memory (Flash chip1 through Flash chip 16) interfaced by two flash controllers. In this example, part of the file data is accessed by one flash controller and other parts by the other flash controller. In certain applications the bandwidth shape of the input/output is important such as a streaming media application. In a streaming media application the streaming software running outputs data at a predefined bit rate (often at the encoded data bitrate such as 2 megabits per second). Processing software coordinates the output from multiple flash controllers or multiple Ethernet interfaces on a single flash controller to be jitter free when data is accessed from a different controller or from a different NIC or different APP CPU. For example, the first 10 megabytes of a stream or file is stored in flash chips 9 through 16 and output using Ethernet 4 756, at 2 megabits per second output rate. After the 10 megabytes is output the next data is output from flash controller 1 700 using Ethernet 1 652 and the transition when data is output from flash controller 2 702 to flash controller 1 700 occurs in a synchronous manner maintaining the overall target bitrate of 2 megabits per second and without significant jitter to cause visual or audio artifacts on the receiving client device (not shown). Synchronization data or output stream coordination processing provides for a smooth transition in terms of the output bitrate so that bit rate jitter does not occur when data output switches from one controller (e.g. 702) to a second controller (e.g. 700).

Host accesses, Priority and Access Rights:

Memory or disk space can optionally incorporate data used to manage access rights, prioritize access to data or busses or resources, and to concurrent access to HDD data. In one example, the concept of a super user is envisioned to configure HDD access for the various elements.

Memory allocation and partitioning:

Memory systems and HDDs typically have management data to coordinate the access of data stored in the memory system. For HDDs a File Allocation Table or other disk data is typically used to indicate the file storage layout for the HDD. For flash memory chips file management data is used to map the data stored in the memory chips to file system information allowing flash memory data to be accessed in file units.

This system uses existing memory system allocation data techniques and optionally adds additional data allocating or partitioning the memory to one or more system elements, or managing access rights (read/write) to the memory. This optional allocation, portioning and access rights data will be called extended disk management data. Extended disk management data can be created during provisioning of the hard drive during manufacturing or can be done after manufacturing and can be updated dynamically. In one example, a system supervisor running on one of the elements configures and manages the extended disk management data during initial disk preparation or dynamically during system operation. Extended disk management data can be stored on HDD itself, in flash memory for flash memory systems, or in another memory. One or more of the system elements can access extended disk management data and depending upon the desired system memory access, allocation, and partitioning strategy access to this data can be restricted to one or more of the system elements.

Figure 9 provides one example of a memory manager interface between a Host Bus (bus 1), a CPU (or CP), and a controller (to disk head array). In Figure 9 Access Control bits (and access control logic not shown) control the access to the various memory chips. In Figure 9 two memory chips Memory Chip 1 and Memory Chip 2 have the access control as shown in boxes A, B, and C. Box A shows the access control for Memory Chip 1 wherein only the Host (bus 1) and controller (bus 3 (bus 3 designator not shown)) and this would be indicative of a memory chip dedicated to a host interface wherein the CPUs or CPs would not be able to access Memory Chip 1. Box B shows the access control for one-half of Memory Chip 2 wherein Host (bus 1), Coprocessor 1 (not shown but can be on a shared memory bus or separate memory bus to Mem Chip 2) and the controller have access to the first half of memory chip 2. Box C in Figure 9 shows the memory access control settings for the second-half of

Memory chip 2 wherein only CP 2 and the Host have access, but both require passwords to access the memory. Logic in Figure 9 can be expanded for more elements, more busses, additional memory chips, etc. It is envisioned that other types of memory access control can be implemented to provide private or shared access to different memory regions to RAM, DRAM, FLASH memory in HDD or Flash drive. While Figure 9 shows memory access control and memory management similar logic and control can be provided for any resource in the HDD or SDD system.

Figure 8 provides an example of HDD data buffering in memory showing the access from various elements. In Figure 8 CP 2 performs access 1 and 41 in the memory buffer, CP 1 performs access 2 and 43. The host interface performs accesses 5,8,2,24, and 42. The controller performs accesses 3,4,6,7 through 23, and 25 through 40. Obviously, other access priorities can be implemented.

New drive configuration - A newly manufactured memory system (HDD or flash memory SDD, or other device) will not have any or will have default extended disk management data stored in the HDD. Default extended disk management data is modified to configure the drive for the desired applications, HDD data allocation, HDD application/host HDD data partitioning, HDD data access control, element access control, element access priorities and other data. Any form of configuration processing can be incorporated over any interface including optional initial access login and password data to access the HDD configuration data.

Host configuration - In one example, extended disk management data (HDD access control, access rights, bus/memory priorities, memory partitioning, element association to memory or HDD data, etc.) is performed over host interface from a host computer. In another example, an element accessible over any interface can be used to configure extended disk management data and applications.

Coprocessor configuration - In another example, extended disk management data is performed via one or more APP CPUs using an interface accessible to an APP CPU.

In another example, any system element can program or configure extended disk management data.

Multiple hosts and multiple host interface switching - Figure 3 shows a single HDD with one host interface connected to a plurality (e.g. four) hosts via a host interface switch. Host interface switch shares a single high-speed HDD with one or more hosts allowing hosts to access the HDD. Because HDDs are becoming so large currently

providing 2 to 4 terabytes of data that in many applications this amount of data can be shared with a number of users over a plurality of hosts interfaces or a switched host interface. HDD Interface switch optionally includes a host arbitration wherein access to the single HDD is arbitrated with any arbitration algorithm such as round-robin or priority allocated wherein hosts are assigned access priority. HDD or SSD data can be locked to a Host Interface or can be accessible by more than one host interface, or controlled using other techniques described in this patent application. In Figure 3 a single HDD or SSD contains processing software allowing coordinated multiple access to HDD data from a plurality of hosts. The processing techniques described for managing concurrent read/write access to the HDD disk array can be applied to the HDD with more than one host interface. In figure 3 the HDD Interface Switch manages the HDD access from a number of host devices. HDD Interface switch can be external to a HDD device or internal to a HDD device. In Figure 4 a HDD with four host interfaces is shown. As stated for Figure 3, any processing, management, arbitration of disk described in other parts of this patent application can be applied to the HDD shown in Figure 4.

Application control from external device such as a VOD system manager
Applications running on CPs in HDD can optionally be controlled from an external host (or system in a distributed processing manner) such as a VOD system manager. VOD system manager will provide CP with configuration or session data for delivering VOD content and the CP will stream the content to the subscriber.

Output connectors: Enhanced HDD incorporates additional input/output (I/O) interfaces and supporting connectors in addition to host interface. Additional input/output interfaces and connectors enhance the input output processing of the HDD. Examples of additional I/O include additional host interfaces, Firewire, USB, wireless, Ethernet interfaces, QAM modulated output, and any other type of input/output. In one example HDD includes two host interfaces along with two Ethernet interfaces, however any other configuration of input/output interfaces and interface connectors can be supported.
Coprocessor to I/O or network interface circuitry
In one example, each coprocessor has direct access to an I/O or network interface circuit (Ethernet for example), allowing a coprocessor to have access to the I/O or

network interface circuit without contenting with other coprocessors, CPUs, host interfaces, etc.

This invention also envisions shared interfaces between one or more coprocessors, CPUs, host interfaces to one or more I/O or network interface circuits, in any combination.

Memory / Bus Architecture / Bus Timing:

Coprocessor memory 152 is used to buffer disk data from disk drive magnetic storage for use by coprocessor(s) 150-150N. Current day HDD drives such as SATA 3.5" HDDs in the 1 Terabyte storage range typically include a 32 Megabyte to 64 Megabyte HDD cache memory and coprocessor memory 152 can share this memory or preferably coprocessor memory 152 is additional memory to prevent a smaller cache memory disk buffer because of coprocessor usage from impacting disk performance. When additional memory is added for coprocessors and is not being used by coprocessor the additional memory can be used as additional host interface disk or flash memory storage buffer data.

An advantage of the coprocessor memory 152 is that it will interconnect with Controller 130 and will use high speed memory and a high speed memory bus connection 134. Connection 134 allows coprocessors 150 and controller 130 high speed access to memory eliminating the need to send disk data over host interface 120. In addition, coprocessor 150 can access data at a higher speed than the interface connection 122 because it will be high-speed memory with a parallel data path and will not be constrained by the host interface connection 122 and host processor speed 110 and host interface 120 processing. And, then by having the coprocessors 150 directly access HDD buffer memory (coprocessor memory 152) the coprocessors eliminate additional disk data transfers through host interface 120 and host interface connection 122 to host 110. While not shown, it is illustrative to note that the host 110 will receive the disk data into host via host interface 122 and then host 110 will process the disk data for outputting to a network or another interface. Application processing performed by coprocessor 150 make the extra data movement through the host unnecessary because the coprocessor 150 will access data from coprocessor memory 152 and the coprocessor 150 will run application software such as an RTSP server without processing support from the host, and the coprocessor 150 will output

network data to network interface, without host processing handling the streaming processing.

There will be various different ways coprocessor memory 152 will be utilized by the system including but not limited to the following:

providing coprocessor memory 152 so that controller 130 /coprocessor 150 can read /write large buffers (or chunks) of data reducing the number of data reads or writes that controller 130 has to perform when accessing data on disk assembly 132.

Provide buffer memory space for each coprocessor 152.

Provide application data buffer space for one or more coprocessors.

Store coprocessor instructions: Other buffer or storage related functions including code storage for coprocessors or CPUs, direct memory access data buffering, host-to-coprocessor, coprocessor-to-host, coprocessor-to-coprocessor, and other related functions.

Other controller 130 or host interface 120 related storage

Application data buffering

Application processor code

Providing memory to coprocessor 150 or host 110 or controller 130 or optional interface 160 memory or combinations of any of the above including but not limited to shared memory buffers between various elements.

Memory management for applications:

An optional element of this invention is dynamic memory allocation for host, coprocessor, and CPU. In some applications, once the HDD is configured and running there will be no need for a large host memory buffer and this memory can be allocated to coprocessor usage. Other allocation strategies of memory to various elements is supported, including but not limited to, on-demand, simple allocation, and demand or load based. Optional dynamic allocation provides memory to applications when required and using memory allocation strategy. In one example, if one coprocessor has a high load and a second has less load the coprocessor with higher load can be allocated more system memory. Element loading and application processing can be used as inputs to determine memory allocation strategy.

Single port or dual port memory:

Coprocessor memory 152 can be any form of single or dual or multiple port memory (RAM, DRAM, DDR RAM, SDRAM, etc.). Coprocessor memory 152 can be

implemented with a single memory chip or single memory array or single memory bus or a plurality of memory chips or memory busses or any combinations of chips, busses, memory arrays. In one example, coprocessor memory 152 is used to eliminate the need for disk array data 132 from having to pass through host interface 120 when processed by coprocessors 150. Coprocessors 150 run application processor program code and data loaded into coprocessor memory 152 or other memory (of any kind not shown) via host interface 110, host interface connection to HDD 122, host interface 120, host interface to controller 124 (or optional host interface to coprocessor interface 158), or via program data from optional interface 160, or program data from hard disk assembly 132, or other coprocessor, or other source including but not limited to flash memory or program memory (both not shown). In one example, coprocessor 150 accesses disk data via dual port RAM (coprocessor memory 152) so coprocessor 150 does not have to go through host interface 120 and has higher speed access to disk data. Other forms of memory can be used other than dual port RAM and it is envisioned that large (2 to 64 megabytes or larger) low-cost memory circuits such as DDR memory will be shared between elements in the system using a single or plurality of memory busses or memory access techniques. CP application software, firmware, allocation, protection, access control and configuration program and data can origination or be downloaded from HDD or SDD memory, non-volatile memory, from other elements, over host interface, or over external HDD I/O. A single memory chip, or a plurality of memory chips can be used for storage of any of the controller or application processor program instructions or data or any other memory described herein.

Multiple Memory Paths for higher speed:

Enhanced HDD 200 in Figure 2 or flash controller optionally includes a plurality of memory busses and memory chips to reduce bus contention between coprocessors and host interface. In one example, host interface has its own dedicated memory chip and memory bus connected to controller 130 and coprocessors have their own memory chip and memory bus connected to controller 130.

Coprocessor access only one memory chip or use one memory bus to allow concurrent coprocessor processing without bus contention.

Any circuit design techniques can be added to the enhanced HDD or flash memory or other controller design to minimize memory bus contention between the various elements. Circuit techniques include employing high-speed memory chips, a plurality

of memory chips, a plurality of memory chips and memory busses, associating a memory chip with only one element, associating a memory chip with a unqiue memory bus with controller 130 allowing controller 130 to feed data to host interface or coprocessors without host interface data access contenting with coprocessor access to memory buffers.

Memory partitioning of a new drive: Optional memory partitioning data allocation memory for host interface buffering and coprocessor usage is programmable and can be set at any time including during the first provisioning of a new HDD, or during operation, or after the loading of new applications, or at any other time.

Protected Memory buffers

MEMORY SECURITY

HDD optionally implements memory protection and allocation techniques for CPs and host using any memory protection technique.

RAM can be protected from other coprocessors

Coprocessor memory 152 may also include memory access protection (not shown) wherein coprocessor 150 access to memory is isolated from host interface 120 access to memory preventing the potential leakage of data from one processor (host 110) for example to one or more of the coprocessors 150. Memory access protection (not shown) will allocate memory to coprocessors(s) 150 and host 110 in either protected memory spaces or shared memory spaces wherein host and coprocessors 150 can access each others data. Any form of memory access protection or memory management can be applied to any memory in the system including coprocessor memory 152 including hardware based or software based memory partitioning where portions of the hardware are allocated to only a single coprocessor 150 or host 110. Coprocessor memory 152 optionally includes programmable control logic or software to allow or prevent shared access to coprocessor memory 152. Additionally, controller 130, or host 110 or coprocessor 150 can set or change the coprocessor memory configuration based on any sharing or isolation scheme required for an application.

Optional support for two or more read/write head arrays

Figure 9 shows an enhanced HDD incorporating 1 or more read/write head assemblies (211A, 211B) interfacing to two controller 220 and 222 over controller to read/write assembly interfaces 214 and 215. In Figure 9 the large scale hard disk 210 includes

two separate independent disk head read/write assemblies that operate independently from each other. In Figure 9 Controller 1 220 controls one read/write head assembly independently of the other read/write head 211B. Controller 2 222 interfaces to a second read/write head assembly 211B via interface 215. Read/write head assemblies include stepper motor or actuator to move read/write head to the correct disk or platter cylinder for reading or writing disk data. Current day drives typically only contain one read/write head assembly and this invention optionally adds additional read/write head assemblyes Two or more read/write head assemblies operate independently from each other allowing concurrent disk platter read/write access. Controllers 220 and 222 (and others if there are more than two r/w head assemblies 211A and 211B) can be contained in a single chip along with other components or in separate chips.

In Figure 9 three CPU and NIC circuits are added in addition to the host interface (not shown) allowing for four elements to provide concurrent input/output of disk data stored on disk platters and accessed by two disk read/write head assemblies 211A and 211B.

Because hard disks are becoming so large (2 to 4 terabytes in 2009) multiple read/write heads and multiple coprocessors yield economic benefits because the terabaytes of data can be input/output to a plurality of users via one or more coprocessors and one or more host interfaces. For a VOD server application 4 terabytes can store about 3,000 full-length movies of about 90 minutes each (H.264 compression with standard definition resolution encoded at 2 megabits).
Redundancy: optional redundant components can be included to provide redundant hardware including but not limited to disk redundancy, redundant spindles/head assemblies, redundant controller(s), interface(s), any form of disk redundancy can be optionally supported by CPs providing for fault tolerant disk processing at the HDD level without host intervention. HDD to HDD redundancy processing interconnected via host interface or HDD I/O provides for redundancy without host processing. In one example, a CP on one HDD communicates with a CP on a second (redundant HDD) to implement HDD level redundancy processing.

Other forms of disk processing such as any level of RAID disk redundancy can optionally be supported by CPs.

When multiple disk head assemblies are incorporated within a HDD the second or additional disk head assembly can provide redundant read only, or read write access to the data stored on the disk platters.

Redundant read/write heads

HDD optionally includes read/write data failure monitoring and automatic switching to another coprocessor or head or drive upon failure. Failure monitoring can also occur on host interface. HDD data is accessable by host, CPs, CPU, and external computers via HDD I/O interfaces. In an example where the host interface goes bad, HDD data can be accessed via one or more HDD I/O interfaces using the requisite access control data such as correct super user account login and password information or other access control data.

Memory storage assembly access control logic provides optional access control (access rights) to storage assembly data and optional access control to device memory. One example of access control logic (access rights) follows. Access rights indicators access rights between on or more of the hardware elements in the device. Examples of access rights include but are not limted to the following: access to all, access to host only, access to coprocessor-N only, access list (host, CP1, CP3). Access control logic and provide read, write, or read/write access control to one or more of the device hardware elements (CPs, interfaces, host). Any type of access control logic (in hardware or software) can set and test read/write access rights based on access rights assigned to device control logic or written/read from storage assembly data, meaning that the access rights can be dynamic based on the element (host or CP) writing the data to disk. In addition, storage assembly can have protected sectors that cannot be modified by CP or other elements along with storage array sector security and sector Locking. Any HDD or SSD data can be locked to one or more elements or access can be shared amongst one or more elements and any form of file system data access techniques can be employed. In one example disk memory protection in the form of extended disk management data contains data providing access protection as shown in the table below.

Disk data or flash memory protection	Controller Access	Host Access	Coprocessor (CP) 1 access	Coprocessor (CP) 2 access	Comments
None	Yes	Yes	Yes	Yes	Unrestricted access to all
Host Read/Write (R/W)	Yes	Yes R/W	No	No	Only host can access
Host RW CP2 R only	Yes	Yes R/W	No	Read Only	
CP1 RW Host R	Yes	Yes R-only	Yes R/W	No	
CP1 locked	No	No	Yes R/W	No	Local data to CP1 cannot directly to disk
CP1 Write All read	Yes	Yes R-only	Yes R/W	Yes R-only	
Optional Intrf 160 R/W CP1 R only	Yes	No	Yes read only	No	Only interface can write and CP1 and interface can read
Host Write w/verify with CP1 Read only	Yes	Yes – one time only to verify the write and only for a limited amount of time	Yes read only	No	Host writes data and can verify it after writing for a limited amount of time or only once

The above table gives some examples of optional extended disk management data used for access control for the different processing elements to disk memory data or physical data written or read from the disk itself. In the above table the example can be applied to any memory size or disk data including but not limited to disk data bytes, disk sectors, disk tracks, disk partitions, disk volumes, etc. Access control manages and restricts where appropriate access to disk data or coprocessor memory 152. In another example, each disk sector is locked to an application or host and is referred to as sector locking. While the above table provides an example of this inventions locking sectors to various elements, the above approach can be extended to have access control information or access control data supplied to authorized elements to access HDD data.

Hardware or software access control can also optionally include encrypting the disk data with one or more encryption/decryption keys. In one example, a single encryption/decryption key is used to encrypt/decrypt disk data. In another example, each coprocessor has its own encryption key and the host interface has an encryption key so data is uniquely encrypted with a key only know to the element writing the disk data and can be only decrypted by the element with the proper decryption key which will typically be the element that wrote the data to disk. When elements include their own disk data encryption/decryption key there is a form of implied access control because when elements access data that they did not write the data will not have the correct key to decrypt unless the writing element exported the key to the reading element. Additionally encryption/decryption processing or hardware can decrypt and export data to other elements. In one embodiment of the invention access control in a similar form as described above is added to encryption/decryption of the disk data.

In the above table the partitioning access protection resolution was at the element level (CPU, Coprocessor, Host, etc.). However, task level partitioning / access protection is also supported with resolution down to a task running on a single or multiple elements (CPU, Coprocessor, Host Interface).

Figure 20 at the top in the section marked ‘A’ provides a high-level drawing of a HDD disk sector. Figure 20 in the section marked ‘B’ provides one example of a

HDD disk sector enhanced by the optional access control element of this patent. While Figure 20 is provided for a HDD similar access control processing and control data can be added to SSDs or other types of controllers. In Figure 20 the "access control" section is exploded with an example of access control data starting with an indicator as to whether the section is Protected, followed by Host Read/Write/Other control for this sector. Following this is additional access control data for CP1, CP2, and other elements. Also shown is Read Access control data that is used to indicate whether a password is required to access the sector or whether the sector is encrypted, similar data follows for write access control for this sector. Another data field provides access control for Tasks and a Task ID is shown defining the access rights for Tasks with certain task IDs. Figure 20 provides only one example of HDD or SSD access control, encryption control, password control on stored data and other techniques of providing access control are envisioned. If Figure 20 Sector Data is the data stored on the disk platter section or in the Flash Memory storage and the other data fields shown such as Sync, Sector Number, etc. is data to manage the storage of data on a HDD. Similar management data for SSD would be used in the case the storage system is a Solid State Drive.

Disk data can also be encrypted with encryption keys unique for a particular element or with keys shared by groups of elements. For example, a pair of coprocessors are running in a redundant manner using encrypted hard disk data and both coprocessors will contain the appropriate decryption key to decrypt the encrypted disk data. Disk data can be encrypted using any granularity (such as one byte, or an entire sector, or entire disk), or raw disk data (such as information locking a sector to a particular element) or combinations. Encryption prevents casual thief of HDD drive wherein a thief attempts to read raw disk drive data because the data is encrypted for a coprocessor or for the host. Any form of access control to access and encrypt/decrypt disk data can be employed to encrypt disk data or raw disk data or combinations thereof. Disk or data encryption can be managed on an exclusive basis wherein a single element has exclusive access to a piece of data and the associated encryption/decryption keys, or where more than one element (CPU, Coprocessor, Host, Flash Controller, tasks running on any of these entities, including applications running on any elements) has exclusive access to data including the security encryption/decryption keys for that data.

Access control can also lock elements to each other, or associate interfaces with coprocessors or other elements. For example, optional Ethernet interface 160 number one can be associated with or locked exclusively to coprocessor 150 number two and optional Ethernet interface number two can be associated with host interface 120 allowing host 110 access to the HDD Ethernet interface 160 (not individually shown).

System optionally includes one or more managed memory busses to maximize efficiency of memory busses and coprocessors and Network Interface Chip (NIC such as Ethernet NIC)) interfaces. There may be one or more head assemblies, one or more memory busses, one or more host interfaces, one or more NIC chips, and one or more coprocessors. An optional circuit arbitrates access to read/write heads, or memory, or both, using any arbitration method including those discussed in other areas of this patent application and priority arbitration where access is prioritized to a host, or cpu, or coprocessor using any prioritization techniques.

Input/Output data rate timing, Timed disk data / stream output
Metered output bit rates for coprocessor or host interface

Coprocessor optional software paces (bandwidth shapes) the coprocessor network output to assure bandwidth spikes do not occur. Depending on the application being supported by the coprocessor input or output bit rate management will be in bits-per-second per stream or packets per second or some other metric. Coprocessor software pacing balances input/output to a plurality of users accessing data so that overruns, underruns, network data overload, and other problems do not occur. For example, if coprocessor is running a video server application with data encoded at 2 megabits per second, coprocessor when outputting file will not output the file at maximum output of the I/O interface (for example, 1 gigabit for Ethernet), but will rather output the data in a streaming application at 2 megabits per second. Likewise in file download applications, coprocessors will share the available output bandwidth amongst multiple users.
Another optional element of this invention is to store contiguous sectors into large disk buffers so that large amounts of data can be written or read at any one time limiting the amount of head positioning time and maximizing data read/write times. This invention also includes processing to reduce the effects of head movement for mechanical HDD and memory chip parallel architecture layout for SSD. For

mechanical HDD excessive head movement will reduce the total number of streams a single HDD will support. Because of this head movement time and coprocessor buffer sizes need to be carefully calculated.

An example calculations for a HDD with a single Ethernet NIC is as follows:

Desired output per coprocessor/NIC: 800 MbPS (Megabits per seond)

Assume HDD has an access speed of 200 MBPS MegaBytes per second = 1.6 Gbps

Assume the HDD head positioning time = 15 milliseconds allowing for 66 accesses per second with time available to read or write only about one sector of say 4 to 16 kbytes.

Reading only 16k bytes per access would allow for only about 1 megabytes of throughput because the movement of the disk head is consuming all the time.

So to maximize coprocessor output of drive and maximize the number of output stream coprocessors minimizes the head movement by using very larger buffer sizes to minimize the time consumed by head movements and rotational latency.

Assume desired goal of 400 streams of output at 2 megabits per second per stream.

1 second of video at 2 Mbps = 250KBps

400 seconds of video requires 100 MB

40 seconds of video requires 10 MB

80 seconds of video requires 20 MB

1 Gigabyte / 400 streams = 2.5 MB

2 Gigabyte / 400 streams = 5MB = 20 seconds of video

Example processing strategy:

Read 20 seconds of video per disk access

400 reads occur every 20 seconds = 50 millisec per read of 5MB

Time to read 5MB at 200 MB per second read time = 1/40 of a second = 25msec.

Time to position head = 15 msec.

Total time = 40 msec therefore 25 buffers of 5MB each can be read per second.

In 20 seconds 500 buffer reads can occur assuming bus is 100% dedicated to reads.

In this case of one NIC outputting the video only a single memory bus would be needed.

A coprocessor system manager can be included on a coprocessor, on the CPU in the HDD, or external to the HDD to perform load distribution where dynamically the coprocessors processing a single stream can be switched from one coprocessor to a second or from on drive to another to reduce the load/wear on a drive.

Co-processor read/writes

This invention includes support for coprocessors to access HDD data. HDD data access by coprocessor can be based on coprocessor including processing software to interpret HDD low level data, or HDD includes interface to allow coprocessors to read and write data stored on HDD. In one example, coprocessor reads HDD file allocation table (FAT) or equivalent. In deployments where coprocessors can write or read files any form of file locking or file semaphores can be used to coordinate multiple concurrent access to HDD data including FAT information, low-level HDD data and file data from host, coprocessors, CPUs and other elements.

Depending upon disk file system and HDD access control, coprocessor may include security control data to allow coprocessor to access low level HDD data. In another example, HDD interface for coprocessor includes access data (access token, password, access security key, or similar), or has access to keys or encryption/decryption support to encrypt/decrypt data.

Coprocessor processing software or HDD supplied interfaces (from controller or CPU or other element) allow coprocessor to read and write file system to access HDD data. In another example, coprocessor includes software to read low-level file system directly from the HDD contoller.

Coprocessor understanding of low-level data format

When Coprocessors understand format of disk data coprocess will include appropriate software for OS used to format HDD data. Coprocessor software optionally includes file system support for fat16, fat32, ntfs, mac, or other file system so coprocessors can directly access low-level HDD data. Coprocessor optionally includes data to processs FAT/Disk/Volume/Partition data and structures and detect changes therein.

Disk data access and arbitration

During HDD operation various elements will make changes to HDD data such as adding new files or deleting files. Optional HDD FAT or disk management data or SSD FAT data or memory management data will coordinate changes to disk or flash memory management data with coprocessors and other processing elements assuring that CPs and other elements update their data when this data changes. In one example a signal is sent to CPs and other processing elements that the FAT has changed, in another example a message is sent that indicates what data is changed. Processing on HDD maintains correct data for all processing elements (Host, CP, CPU, etc.) or

signals processing elements when data has changed allowing processing elements to update their data for the changes that occurred.

Expand upon dynamic allocation and what happens in the host when the fat is updated on the drive and with the coprocessors, etc. CPs and/or host need to know sector data has changed with some form of file locking.

Coprocessor accesses, Priority and Access rights:

Coprocessor optionally processes raw disk data to/from coprocessor memory 152 offloading this task from controller 130. During HDD read operations for example, controller 130 retrieves raw HDD data to coprocessor memory 152 that includes raw HDD data (HDD header data, sector data, error-correcting control bits, etc.) along with application specific data fields. In some cases when HDD data is sent to host processor 110 controller 130 will remove hdd related data and send only application related data (without raw hdd data fields) back to host. Example, controller 130 reads 16 megabytes at one time for one application that includes HDD related data that would normally not be transferred to host 110. Coprocessor 150 includes processing to ignore or process and filter out the HDD related control data and error correction data and select only the application data that would normally be sent to host 110. This processing eliminates having to have controller 130 properly process the raw disk 132 data into host equivalent data and coprocessor 150 retrieves only the data of interest (application layer data) skipping raw HDD related data that would normally not be sent to host 110. This processing would not be necessary if controller 130 can skip raw HDD related data and return only application data to coprocessor memory 152. During HDD write operations coprocessor 150 can add necessary raw HDD related data to application data being written to disk. In yet another example, coprocessor 150 apply application level processing and formatting to disk data during reading of disk data or when writing disk data. An example of this is when coprocessor 150 or other element writes data to disk that includes TCP/IP IP header and framing information so that when coprocessor 150 or other entity access this data IP framing processing does not need to occur on the data.

Similar to the access control techniques described herein, various techniques to partition disk data can be supported including static or dynamic access control partitioning. In static partitioning the disk partitioning does not change much and the disk partitioned remains constant for a long period of time. An example is to partition

a portion of the drive for host 110 write access and coprocessor 150 read only access. The host writes data to the disk partition and then the coprocessor 150 can read that partition. In this example the coprocessor 150 would serve the disk data that the host 110 wrote into the partition. Partitioning information includes access control data can be written to disk as part of raw disk data or store in another memory separate from raw disk data.

Dynamically partitioned disk data is data partitioning that changes frequently such as in a file server where files are rapidly added and removed. In such a system the disk portioning will be dynamic and coprocessors and other elements will use dynamic partitioning methods for managing disk data usage.

Coprocessor applications:

Application programs run from RAM, or Flash or from application store on device include an application processor instruction memory loader appropriate for the type of program memory. When program memory is non-volatile memory such as Flash memory loader is a Flash memory loader. For DRAM or RAM memory, loader loads program into DRAM or RAM memory containing instructions executed by coprocessor.

Optional code signing techniques verify the integrity of the code to only allow code with proper digital signatures to execute on the CP. CP applications can be stored in non-volatile flash memory, or on the HDD or downloaded from host or over CP I/O interface or from CPU or other entity. CP application memory and CP data memory can be optionally partitioned or protected from other system elements.

Boot Process:

There is a wide range of techniques that are envisioned by this patent application to control the startup, execution, and management of coprocessors and the application software or firmware running on the coprocessors. Coprocessors can automatically start to run when the HDD system is powered on they can be controlled from another element such as from the host via host interface or from the controller or CPU or another coprocessor or from special logic (not shown). In one example coprocessors automatically look for coprocessor specific application software in a particular area of memory or from HDD data or from the HDD using an application loader. In another

example, coprocessors sit in an idle state until released from idle using a control signal. In yet another example, an external signal or message, or interface controls the execution of an application.

Application code for coprocessors is loaded into a non-volatile memory chip or onto the HDD drive platters as HDD data or externally loaded from an external interface such as the Ethernet interface or host interface. Software or firmware update

Application running on the coprocessor is optionally contained to certain regions of memory or certain areas of the HDD low-level data via access control circuitry. In another example, coprocessors are prevented from writing file allocation table data and can only read file allocation table data. In yet another example, one coprocessor is configured as a system master and has read/write control of the HDD while other coprocessors only have read level access as configured by another element.

COPROCESSOR LOCKING TO PREVENT OTHER APPS FROM BEING LOADED:

This invention also allows coprocessors to be locked into an idle state while certain processes are taking place such as the formatting of a HDD or running HDD maintenance applications such as chkdsk preventing the coprocessors from attempting to access data that is changing on the HDD.

Coprocessors and application code running on the coprocessors have access to data buffer memory for the coprocessor. In one example, coprocessors memory is protected from access from other elements. In another example, coprocessor memory regions are allocated to coprocessors or element to prevent one coprocessor or element from corrupting the memory space of another coprocessor or element.

CP applications can use any techniques to update CP applications stored on HDD, SSD, or in separate CP program memory space.

CP and Host interaction:

In one embodiment of the invention a host can interact with coprocessors or CP resources such as CP memory using extended host to device interface commands. For example, a SATA Interface, or Flash Device interface has device control specific commands used by the host when interfacing with the HDD or device. Extensions to

these commands are envisioned to allow the host to interface with the CPs including extensions for the following:

configuring CPs, memory, low-level data access, associating CPs with interface chips, memory bus access priorities, CP access control data (access rights, access tokens), and other CP control functions.

Obtaining CP status, starting/stopping/resetting CPs, including CPs determining host status.

Loading new application code for CP in CP application storage (on HDD, or in non-volatile memory, or similar)

Host to CP sending or receiving messages. CP to CP sending or receiving messages. CP to external I/O interface sending and receiving messages.

Configuring CP or other element related data such as IP addresses for Ethernet stacks running on CPs, or other configuration or I/O related data associated with other I/O interfaces on HDD. Or configure CP or I/O element or other element for network or I/O parameters such as IP addresses or using DHCP network protocol for IP address management.

One or more the CP controls described herein or normally associated with controller a process are integrated into application processor control logic used to control and monitor the application processors and includes but is not limited to resetting the CP, detecting CP status, setting CP status, starting and stopping applications, and other status or control functions.

Coprocessors run portions or entire protocol stacks to support the processing needs of the I/O interface associated with a CP.

Extended API to interface host with Coprocessors

One or more different types of interfaces can be provided to allow the host access to CP or HDD status and configuration data. CPs can also have interfaces to obtain status or coordinate CP and Host distributed processing. Host can interact with CP over host interface or over any other I/O interface supported by HDD. In one example, host to HDD interfaces (Application Program Interface such as SATA Interface data/commands) are extended to allow two way communications from host to any HDD element including CPUs, CPs, I/O Interfaces, CP memory (program or data), HDD configuration data, etc.

Coprocessor Control:

Logic to control coprocessor operation from a host or main device processor or interface includes control logic such as holding a CP in idle, or resetting an individual CP, or enabling a CP to switch to the running mode, or having the CP sleep or block waiting for a message is incorporated. In one example a CP software boot loader reading CP application software from the HDD or SSD holds the CP in idle until the software is loaded and after software is loaded in the CP application memory space changes the CP control logic enabling the CP to run and execute code.

Application data buffers:

Applications running on coprocessors 150 can be stored in flash memory allocated to the coprocessor 150, or can be loaded into coprocessor memory 152 or in other memory (not shown).

Application program buffers:

HDD memory of any type disk, flash, or RAM can be allocated to general HDD usage or allocated to one or more of the system elements.

Download of code to host processors:

An application program downloader is added to the HDD to allow programs to be loaded for execution by coprocessors 150. Any form of application downloading or flash memory programming for storing applications can be used to load application program code (binary or Java or Basic or scripting or other) for coprocessors. Application program code for coprocessors 150 can originate from any element or host or interface including from the hard disk assembly. A coprocessor 152 program loader will load the coprocessor's program into an appropriate memory location for execution by coprocessor 152. Any form of access control or digital signature checking on the application code being downloaded to the coprocessor 152 can be applied to the downloader or coprocessor software boot process. Optional application management security provides access control to applications including requiring a security key or authentication to load an application, requiring a security key or authentication to start or stop an application. Any form of security authentication, challenge, or cryptographic techniques can be used for one or more of the elements of this invention. It is envisioned that a new category of programs called DiskApps™ will be generated to run on the HDD coprocessors 150. 5. In one example a Public/Private key chain is used to authenticate a digital signature contained in the CP

application program before loading the code verifying the digital signature code to a known root of trust such as a device manufacturer or software developer with a digitally signed software developer root key signed by a trusted authority.

Application firmware updates

CP application software updates can be managed or unmanaged. In managed application software updates a system manager or super user updates CP application software in flash memory or on HDD while HDD is idle or held in reset or not running. Access control is optionally applied to prevent applications from hijacking the CP and performing malicious processing. In one example, Application firmware includes digital signature checking to verify signature on code image before running. In another example, secure login to firmware update process must be performed before firmware can be modified. Other security techniques can be applied to CP application software, firmware, configuration and data updates.

One or more watchdog timers are used to recover a hung or crashed CP. Watchdog timer techniques can be added to any coprocessor applications, coprocessor controls (reset, status, etc.), for CP to other CP or main device CPU or host to provide recovery hardware from CP or CPU crashes.

HDD resources such as watchdog timers, DMA controllers, secure clocks, and other hardware elements can be shared amongst the elements or allocated to one or more elements using any from of access control.

Coprocessor access to other resources on HDD or on host or external computers: It is envisioned that CPs can have optional access to HDD resources along with resources on an external host or even resources on external computers connecting to HDD over host or I/O interface.

Content Ingest (PVR recording and bus timing) example: HDD bandwidth is reserved for nPVR recording, Ingest and building index files for Personal video recording PVR. For PVR multiple index files may be created and in one example the following speed index files are crated and stored: 4x,16x, 32x index files. Creation and storage of index files and storage or input content is managed so that numerous ingest seek operations (content ingest plus index file creation) do not impact main device operation because the hdd is only good for 60 seeks per second. In addition PVR

trick-play file playout includes large data buffers minimizing the number of read operations thus maximizing the number of streams that can be output by minimizing the head movement.

Coprocessor 150 is any form of central processing unit (CPU), CPU core or cores, bit-sliced processor, state machine, or processing device that performs digital or computer processing. Examples include but are not limited to ARM 11 CPU cores licensed by ARM, Power PC cores licensed by IBM and others, Intel CPU chips such as the 8051 or 8086 or Pentium or Xeon CPUs, and can operate on any bits size of data from 1 bit to over 2048 bits with 8 to 64 bits being typical. Coprocessor will execute software instructions from a coprocessor instruction memory (not shown) or coprocessor ram memory (not shown) or from coprocessor memory 152. The term coprocessor will be used to describe any form of CPU including one or more cores in a multicore CPU. Any application or portions of an application or application assistance can be run on coprocessor(s) 150 example include but are not limited to:
tcp/ip and network processing sending/receiving/managing hdd data
network file system processing sending/receiving/managing hdd data
video-on-demand (VOD) server receiving and storing video data and then outputting stored video data in any one or more formats (unicast, multicast, adaptive streaming, RTSP, UDP, RTP, etc.)
file server protocol control processing such as fast-forward, rewind, etc. for video-on-demand or Personal Video Recorder (PVR) functionality
Real Time Protocol Server (RTSP), Video stream adaptive streaming protocol software, Software encryption or cryptographic processing of HDD data, HDD disk array processing (RAID or SAN or other) without host processing or with assistance from host processor
RAID processing performed by HDDs without assistance from host or or with assistance from a host controller.
Providing multiple access paths to disk data in the event one of the interfaces or the primary host interface is damaged
Secure data access to HDD data including but not limited to processing access control list for HDD data
Any other from of application running solely on disk or an application split across a host and one or more disks

Host-less disk redundancy wherein two or more HDDs provide redundancy without a host controller.

Coprocessor to coprocessor distributed processing with coprocessors communicating on one or more drives to provide distributed processing

CP to host or other processing wherein CP performs a portion of the processing in a distributed processing environment.

Extending host interface to support application data being sent over host interface in addition to normal HDD host data and commands. An example includes sending RTSP commands over host interface with routing to coprocessor so coprocessor processing RTSP data.

Targeted Ad insertion where advertising material targeting specific users is inserted or spliced into content.

Broadcast Ad splicing where advertising material is spliced into content

An example of application processing for a VOD/PVR server will be provided in the example below:

Optionally partition memory statically or dynamically and set optionally assign access rights to disk data or memory for each coprocessor and host in the system. This memory will be used to store a movie file. Partitioning can be performed by controller 130 or host 110 via host interface 120 or coprocessor 150 or an external host connecting over optional interface 160. Partitioning may have been performed during disk formatting.

An element in the system (host 110, CP 150, Controller 130, other interface connection 160) writes a movie file to disk. Disk memory may have been partitioned in the step above, or store movie file on disk (without coprocessor specific partition).

Control program for CP1 is loaded into CP1 150 instruction program memory (not shown but can be part of any memory including memory 152 or a separate memory). Control program in this example will include a portion of all of the VOD server application code and other code required by CP1 150 to perform its processing such as TCP/IP processing code or interfaces to a TCP/IP stack, or TCP coprocessor, or other Input/Output I/O interface or I/O coprocessor.

CP1 150 begins execution of control program (VOD server app code) after an optional check of a code boot integrity check signature to make sure the code has a correct digital signature applied to prevent rogue code from being loaded into CP1.

A command is sent to the application processor and in this example it is CP1 150.
CP1 150 processes command that includes the name of a file to play and other configuration data such as IP address port number and interface to use for outputting data such as optional Ethernet interface 160 or host interface 120.
CP1 150 performs a disk data read request to controller 130 via command interface memory in coprocessor memory 152 or via optional coprocessor to controller interface 156. CP1 150 performs a second disk data read to fill double-buffering of movie disk data.
Controller 130 reads disk data into coprocessor memory 152.
CP1 150 processes disk data in memory 152 and begins outputting the data to Ethernet interface with RTP and UDP encapsulation of MPEG compressed video data.
CP1 150 continues to output stream until an end-of-buffer condition and then switches buffers, requests an additional buffer read to controller 130 and monitors control signal inputs from other elements. In this example, RTSP data is sent over optional additional interface 160 Ethernet connection and CP1 processes received RTSP data over this stream along with other data such as adaptive stream bit rate settings, etc.
CP1 150 continues repeats steps 'f' and 'g' above until entire movie is output or a command instructing the coprocessor to perform different processing is received such as an RTSP stop-play or rewind or fast-forward or restart from beginning is received.
In the above example, and any other example in this patent application CP1 150 optionally processes disk data before storing to disk or before outputting from disk.

In the above example RTSP commands were sent by an external device to CP1 150 via optional Ethernet interface 160. Application and other data and commands can be communicated with any element in enhanced HDD 200 using any interface using existing HDD/interface interface communications techniques or new communications techniques to communicate with elements in HDD 200 over any interface. An example is to extend ATA/SATA or SCSI Application Program Interface commands or interface commands to include data to or from any element in enhanced HDD 200. Extension of ATA/SATA/IDE or SCSI Application Program Interface will contain address information to address one of the numerous elements added to a HDD by this invention including but not limited to address information for one or more coprocessors 150, one or more optional interfaces 160, coprocessor memory 152, etc.

Another example is where a DVB Simulcrypt interface is running on a CP performing content scrambling (encryption) and interfacing to an external security system following the DVB Simulcrypt protocol. Other forms of content encryption and access control/conditional access security can run on one or more of the CPs.

Optional additional interfaces 160 or host interface 110 may also include a Conditional Access System (CAS) interface or DVB Simulcrypt interface to allow coprocessor 150 or controller 130 to encrypt data using conditional access techniques or video encryption techniques as described by the DVB organization in the DVB simulcrypt specifications.

Coprocessors will scramble video content (or any data) using a random number and then feed that random number to an ECM generator as part of a DVB simulcrypt interface. Coprocessor can optionally provide support for a DVB Simulcrypt Synchronizer. Other video or content conditional access security processing or Digital Rights Management (DRM) processing can be added to coprocessor.

Enhanced HDD 200 in Figure 2 may contain a plurality of interfaces including one or more host 120 interfaces and one or more additional optional interfaces 160.

Disk2Disk D2D sharing of load with partial content coming from multiple disks (scheduler schedules a portions of content from a plurality of disks)

For high throughput systems HDD data can be replicated across multiple HDDs with complete or partial replication of HDD data. In one example, four HDDs are used and each one stores one-quarter of the content file. For the complete output of the file all four HDDs need to provide their portion of the stored content. Coordination of the file content output from the four HDDs appears seamless as a result of optional file processing software coordinating the activities of the four different drives resulting in what appears as a single contiguous output stream of the file. Coordination at the HDD level and at the network address and data processing level is incorporated in CP or host or CPU or other element software or firmware.

Flash memory data or HDD data can optionally be stored content across a plurality of SSDs or HDDs. When content data is stored on a plurality of SSDs or HDDs the

content input/output is coordinated to eliminate video artifacts due to jitter, and the input or output appears to the receiving or sending device as though it is coming from or going to a single device. CP or other element processing coordinates the input or output of content stored across multiple SSD or HDDs for seamless operation.

Application security

CPs and other processing elements optionally incorporate any one or more security methods to secure the CP application, CP application execution, CP application software updates, CP configuration data, CP to host interface, CP or other element access to/from the host, host access to/from the CP or other elements, CP operation, access to HDD data, and for other requirements.

CAS to prevent the theft of a HD from yielding hundreds of movies

Any form of HDD or SSD content protection can be added to protect the content. Because HDDs are becoming so large (multiple terabytes) a single HDD can store thousands of movies or video content. Security in the form of encryption can be added with a secure encryption key manager either on the HDD itself (a secure micro or security coprocessor added to the HDD controller or external to the HDD).

Store CP data in application format.

HDD or CP data can be stored in a preprocessed application ready format to eliminate the need for CPs to add the processing on each input/output stream. For example, with UDP framed or RTP/RTSP framed video data the HDD data will include UDP and RTSP/RTP framing so CP can output RTSP/RTP streamed video in correct format from HDD or SDD without added processing.

In certain applications where the network or other data processing that must be performed on the Hdd content data is fairly static, the HDD data can be preprocessed to store the data in an input or output compatible preprocessed format so that the HDD data does not need to have the fairly static processing performed every time the content is input/outputted.

Claims

I claim:

1. An enhanced storage device comprising a memory storage assembly connected to a device controller, device controller connected to a host computer interface, device controller connected to device memory, an application processor connected to an application processor input/output interface and device memory, application processor control logic connected to device controller wherein device controller controls application processor, and device memory access arbitration logic connected to both device controller and application processor wherein device memory arbitration logic arbitrates device memory access between device controller and application processor, and an application processor instruction code loader connected to device memory wherein application program code is loaded into device memory.

2. Claim 1 wherein device memory is comprised of a plurality of memory chips and a device memory bus interconnecting memory chips to device controller and application processor.

3. Claim 1 wherein enhanced storage device includes memory storage assembly data access control logic controlling memory storage assembly data accesses from host computer connected via device controller host interface and application processor based on access control bits stored in memory storage assembly.

4. Claim 1 wherein enhanced storage device includes memory storage assembly data access control logic controlling memory storage assembly data accesses from host computer connected via device controller host interface and application processor based on control data processed by device controller.

5. Claim 1 wherein application processor instruction code loader verifies a digital signature code authenticating application processor instruction program.

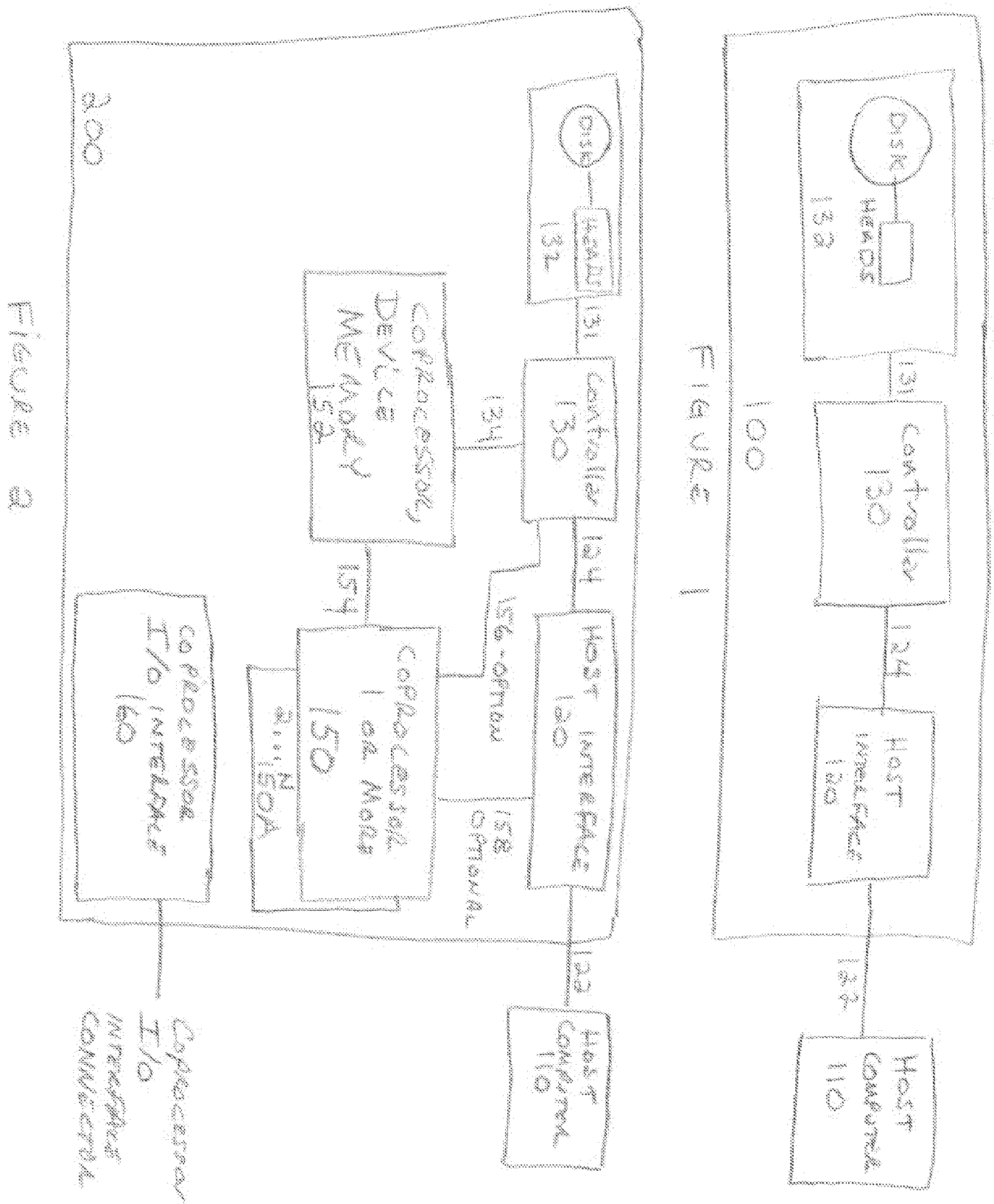
100
Disk
Head
132
131
Controller
120
124
Host Interface
122
132
Host Computer
110
Figure 1
200
Disk
Head
132
131
Controller
130
134
Coprocessor Memory
152
154
Coprocessor
150
Coprocessor I/O
160
156
158
Optional
Host Interface
120
122
Host Computer
110
Coprocessor I/O Interface Connection
Figure 2

Figure 3 Host Interface Switch

Host Interface Switch HIS includes optional arbitration.

Host 1 thru N can be remote host over interface (SATA, Ethernet, etc.) or coprocessor or combination. Coprocessor can contained in HDD/SDD digital ASIC or digital controller chip on the HDD/SDD. HDD can also have multiple HOST interfaces of different types (2xSATA, 1xSAS, 4xEthernet, etc.)

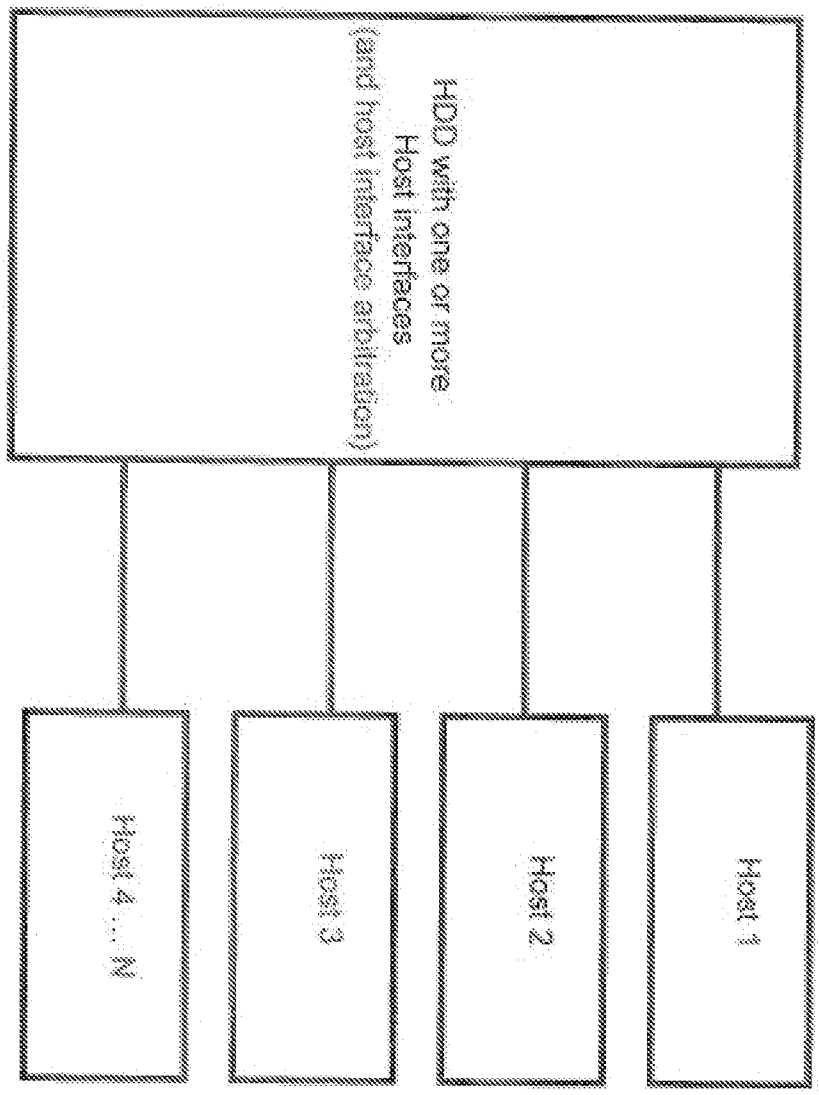

Figure 4

Multiple Host Interface HDD with Host Arbitration on HDD

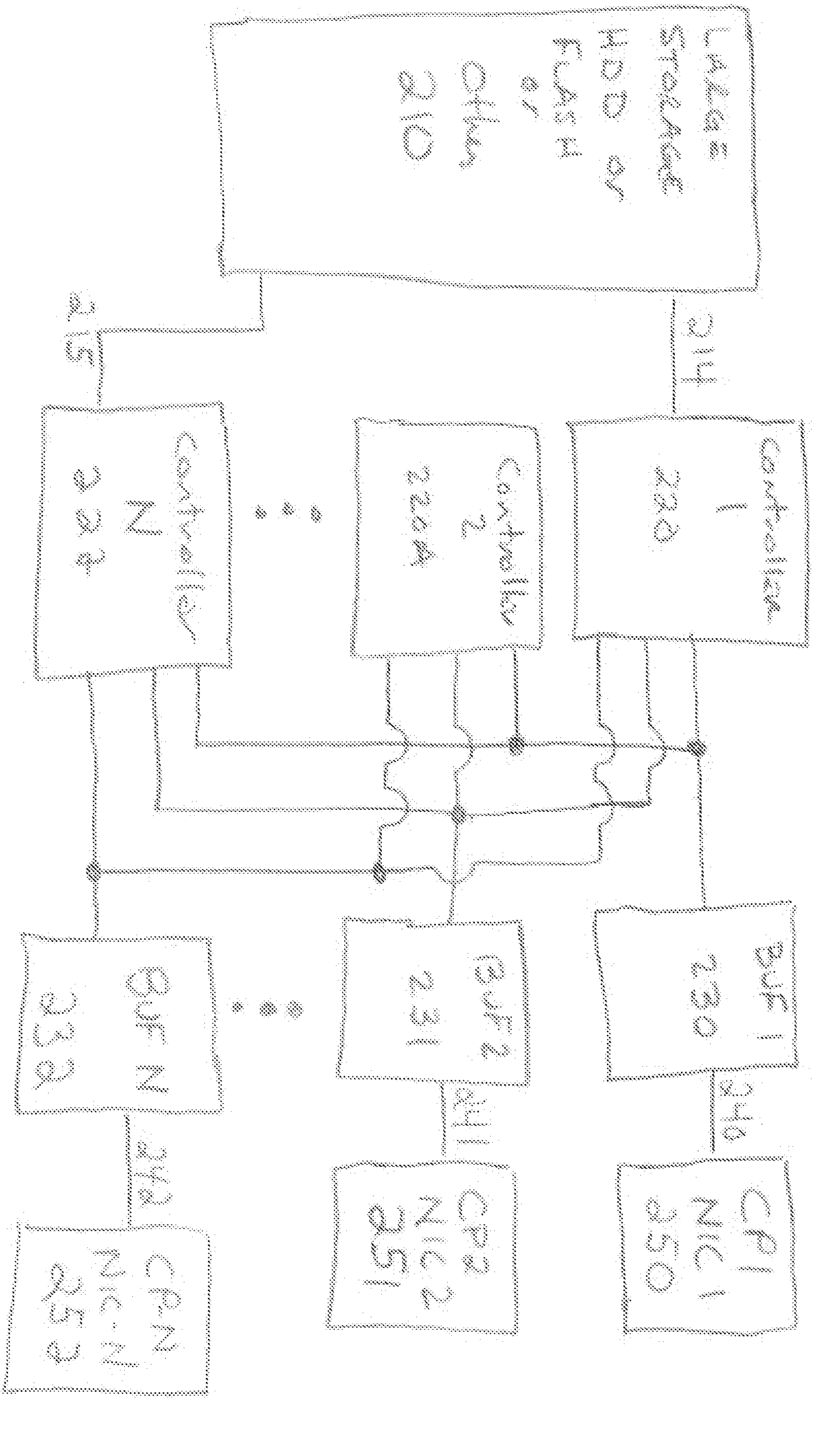
LARGE STORAGE HDD or FLASH or Other 210
215
214
Controller 1
Controller 2
Controller N
BUF 1 230
BUF 2 231
BUF N 232
CP1 NIC 1 250
CP2 NIC 2 251
CPN NIC N 252

FIGURE 5

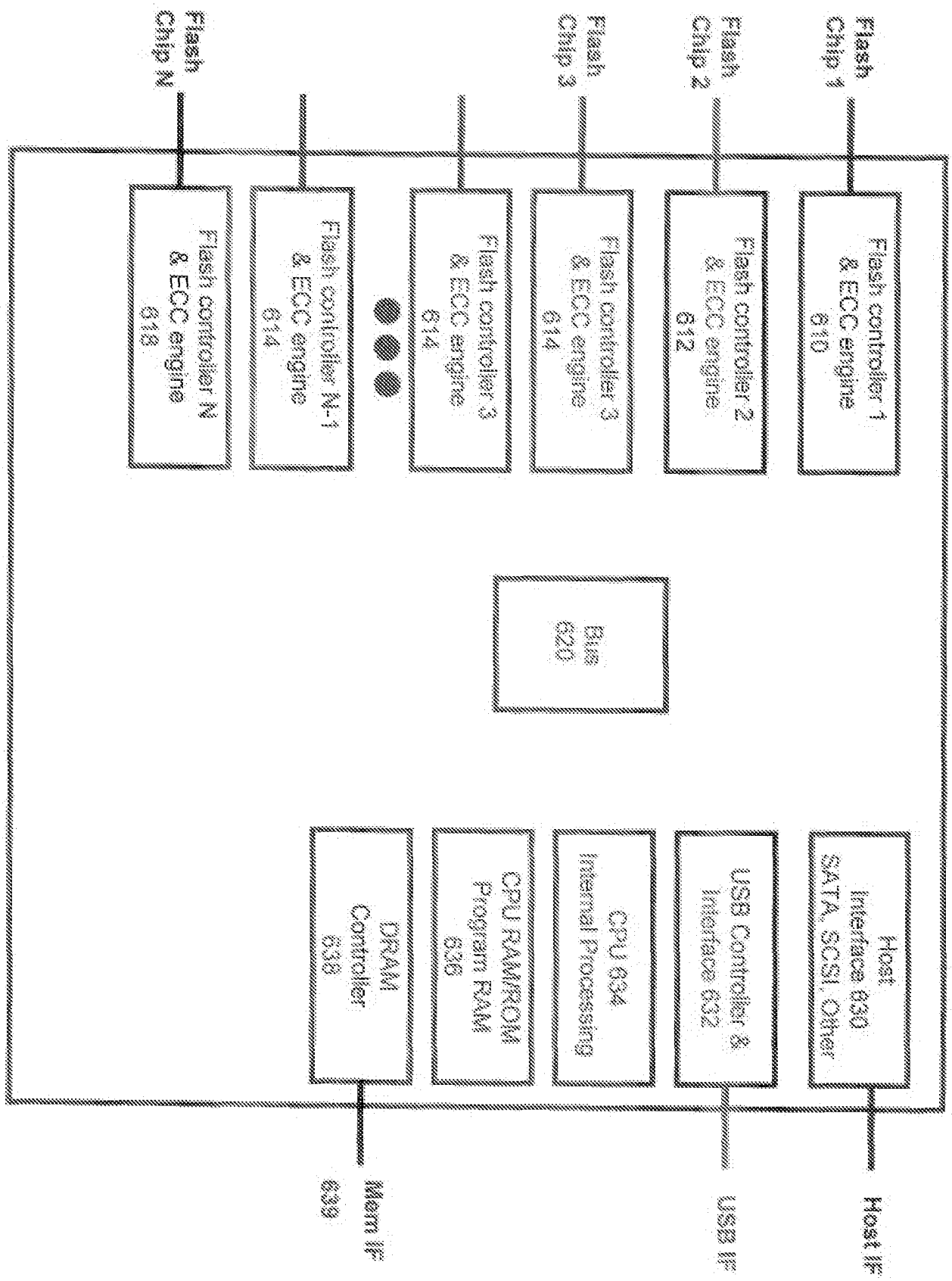
Flash Chip 1
Flash Chip 2
Flash Chip 3
Flash Chip N
Flash controller 1 & ECC engine 610
Flash controller 2 & ECC engine 612
Flash controller 3 & ECC engine 614
Flash controller 3 & ECC engine 614
Flash controller N-1 & ECC engine 614
Flash controller N & ECC engine 618
Bus 620
Host Interface 630 SATA, SCSI, Other
USB Controller & Interface 632
CPU 634 Internal Processing
CPU RAM/ROM Program RAM 636
DRAM Controller 638
Host IF
USB IF
Mem IF 639

Figure 6

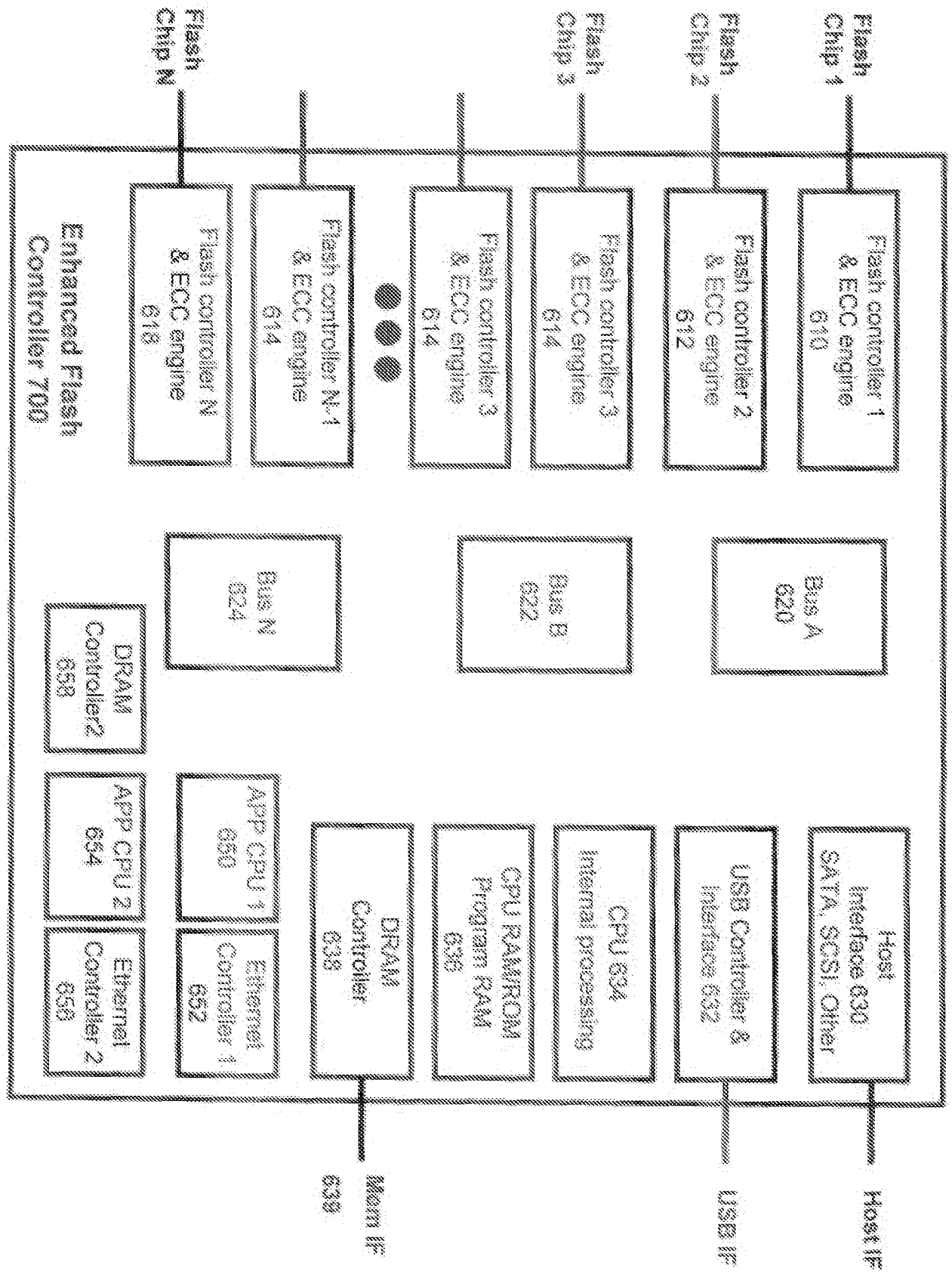
Flash Chip 1
Flash Chip 2
Flash Chip 3
Flash Chip N
Flash controller 1 & ECC engine 610
Flash controller 2 & ECC engine 612
Flash controller 3 & ECC engine 614
Flash controller N-1 & ECC engine 614
Flash controller N & ECC engine 618
Enhanced Flash Controller 700
Bus A 620
Bus B 622
Bus N 624
Host Interface 630 SATA, SCSI, Other
USB Controller & Interface 632
CPU 634 internal processing
CPU RAM/ROM Program RAM 636
DRAM Controller 638
APP CPU 1 650
APP CPU 2 654
Ethernet Controller 1 652
Ethernet Controller 2 656
DRAM Controller2 658
Host IF
USB IF
Mem IF 639

Figure 7

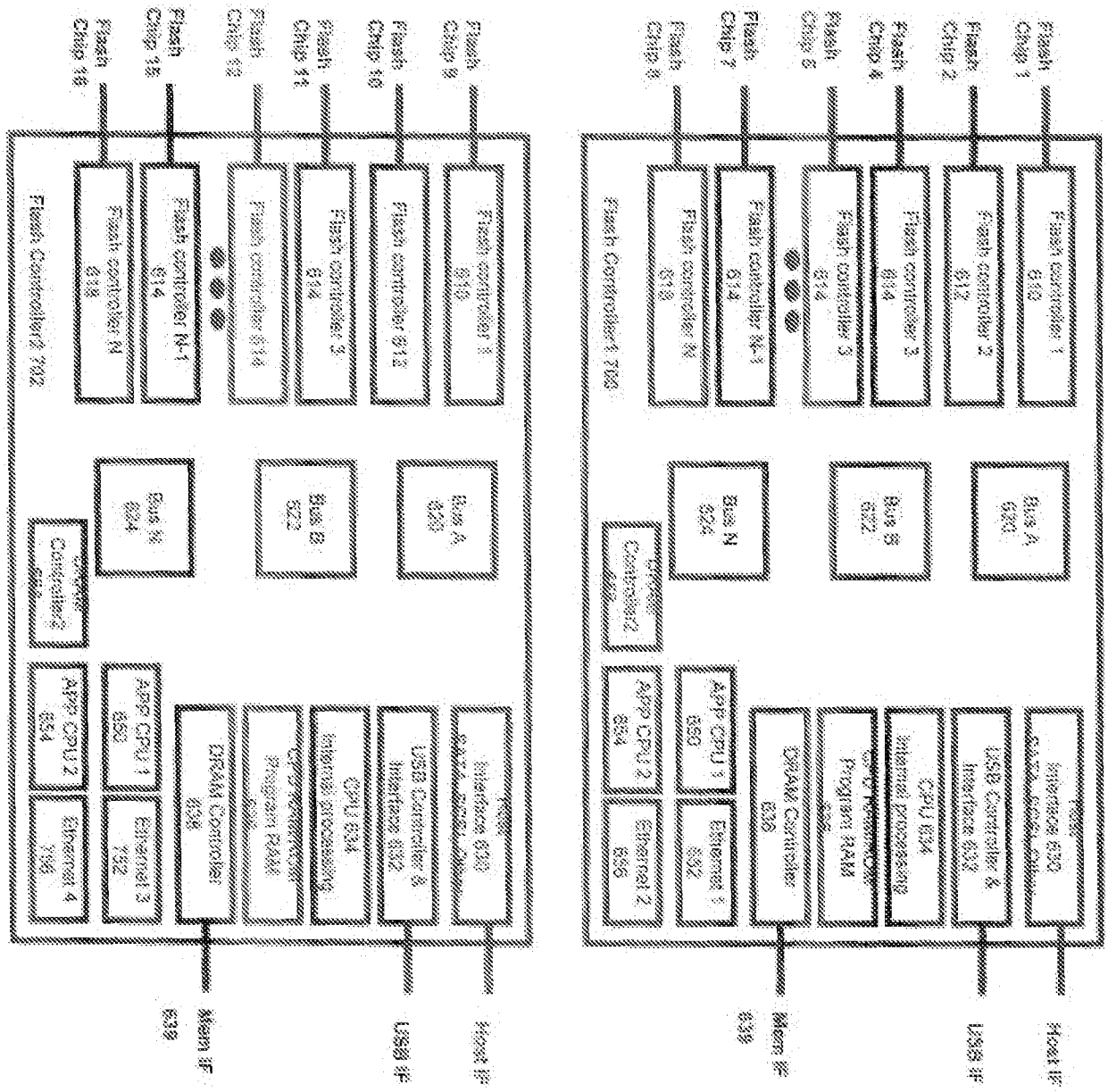
Flash Chip 1
Flash Chip 2
Flash Chip 4
Flash Chip 6
Flash Chip 7
Flash Chip 8
Flash controller 1 612
Flash controller 2 612
Flash controller 3 614
Flash controller N-1 614
Flash controller N 618
Flash Controller 1 703
Bus A 630
Bus B 632
Bus N 624
Controller 2
APP CPU 1 650
APP CPU 2 654
Ethernet 1 652
Ethernet 2 656
Interface 630
USB Controller & Interface 632
CPU 634
Internal processing
Program RAM
DRAM Controller 638
Host IF
USB IF
Mem IF 639
Flash Chip 9
Flash Chip 10
Flash Chip 11
Flash Chip 12
Flash Chip 13
Flash Chip 16
Flash controller 1 612
Flash controller 2 612
Flash controller 3 614
Flash controller N-1 614
Flash controller N 618
Flash Controller 2 702
Bus A 630
Bus B 632
Bus N 624
Controller 2
APP CPU 1 650
APP CPU 2 654
Ethernet 1 652
Ethernet 2 656
Interface 630
USB Controller & Interface 632
CPU 634
Internal processing
Program RAM
DRAM Controller 638
Host IF
USB IF
Mem IF 639

Figure 7A

Example device memory 152 accesses	HDD Disk heads 132 via device controller 130	Host access 124 via controller 130	CP1 access via 154	CP2 access via 154
1				Write
2			Read	
3	Write			
4	Write			
5		Read		
6	Write			
7	Read			
8		Write		
9	Read			
10	Read			
11	Read			
12...20	Write			
21	Write			
22	Write			
23	Read			
24		Write		
25	Read			
26	Write			
27	Read			
28	Write			
29 ... 40	Reads and writes			
41				Read
42		Write		
43			Read	

Figure 8

Example of read/write access and access priority for accessing device memory using elements shown in Figure 2 with 2 coprocessors.

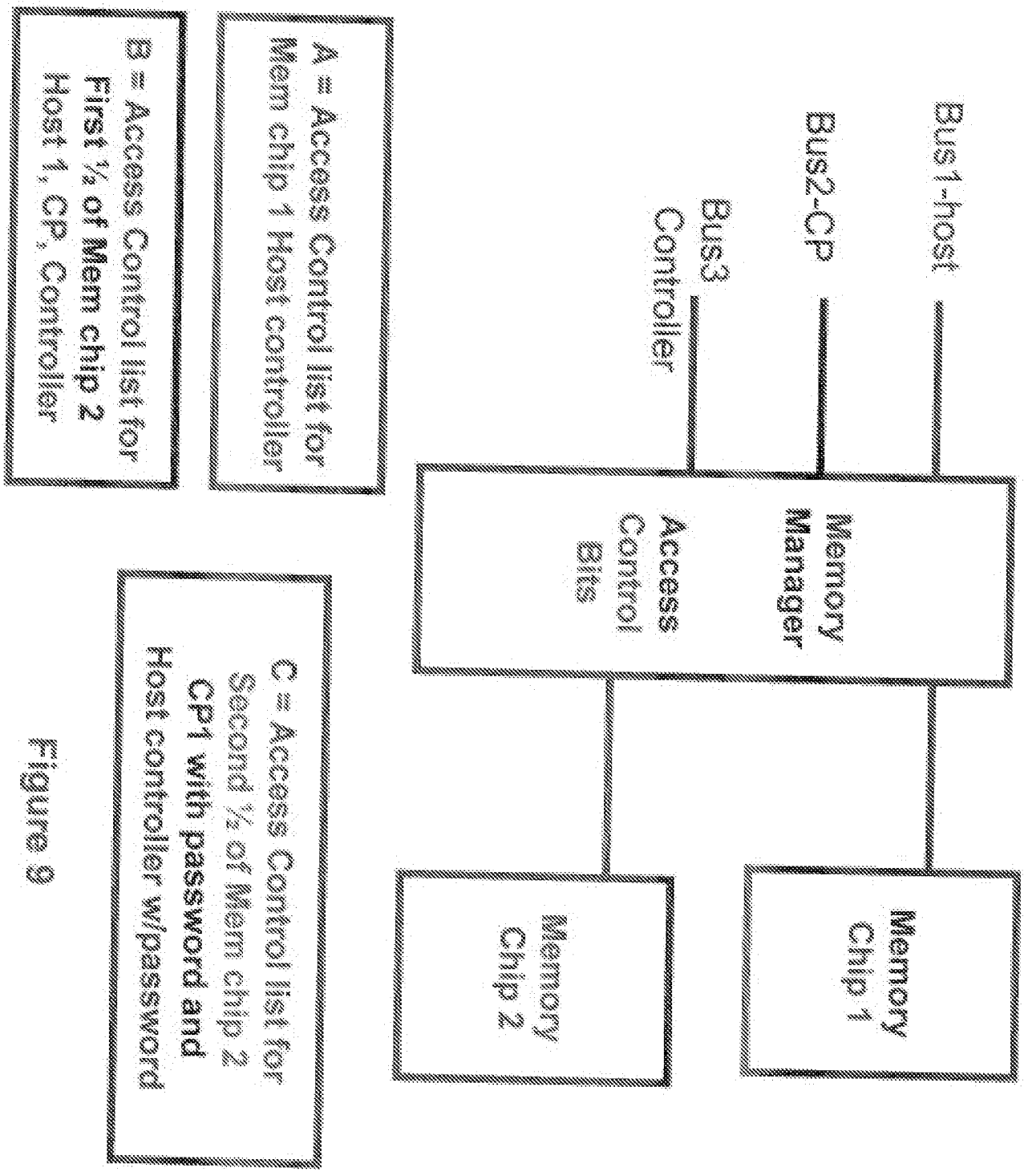
A = Access Control list for Mem chip 1 Host controller
B = Access Control list for First ½ of Mem chip 2 Host 1, CP1 Controller
C = Access Control list for Second ½ of Mem chip 2 CP1 with password and Host controller w/password
Bus1-host
Bus2-CP
Bus3 Controller
Memory Manager
Access Control Bits
Memory Chip 1
Memory Chip 2

Figure 9