US 20080147931A1

(19) **United States**

(12) **Patent Application Publication** (10) **Pub. No.: US 2008/0147931 A1**

McDaniel et al. (43) **Pub. Date: Jun. 19, 2008**

(54) **DATA STRIPING TO FLASH MEMORY**

(75) Inventors: **Ryan Cartland McDaniel**, Hudson, MA (US); **Thomas McCormick**, Chelmsford, MA (US)

Correspondence Address:
PERKINS COIE LLP
PATENT-SEA
P.O. BOX 1247
SEATTLE, WA 98111-1247

(73) Assignee: **SMART Modular Technologies, Inc.**, Fremont, CA (US)

(21) Appl. No.: **11/585,121**

(22) Filed: **Oct. 17, 2006**

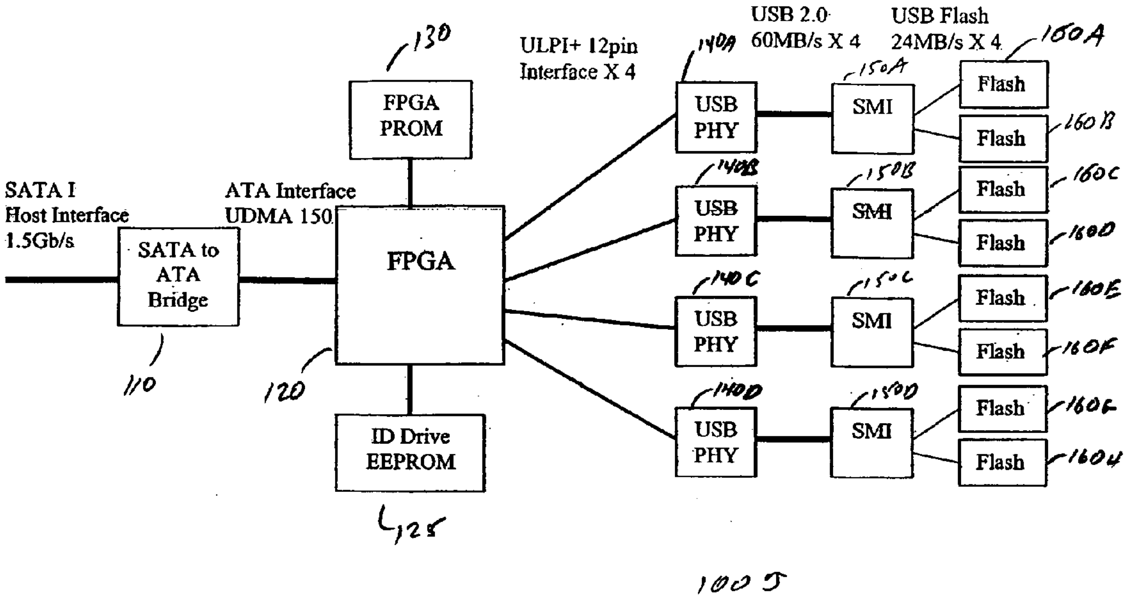
Publication Classification

(51) **Int. Cl.**
G06F 13/12 (2006.01)

(52) **U.S. Cl.** **710/74**

(57) **ABSTRACT**

In various embodiments, options for data striping to FLASH memory are provided. In one embodiment, an apparatus is provided. The apparatus includes an SATA to ATA bridge, an ATA to USB bridge coupled to the SATA to ATA bridge, and a USB interface coupled to the ATA to USB bridge. The apparatus also includes a first FLASH memory controller coupled to the USB interface. The apparatus further includes a first FLASH memory module coupled to the first FLASH memory controller. The apparatus also includes a second FLASH memory controller coupled to the USB interface and a second FLASH memory module coupled to the second FLASH memory controller.



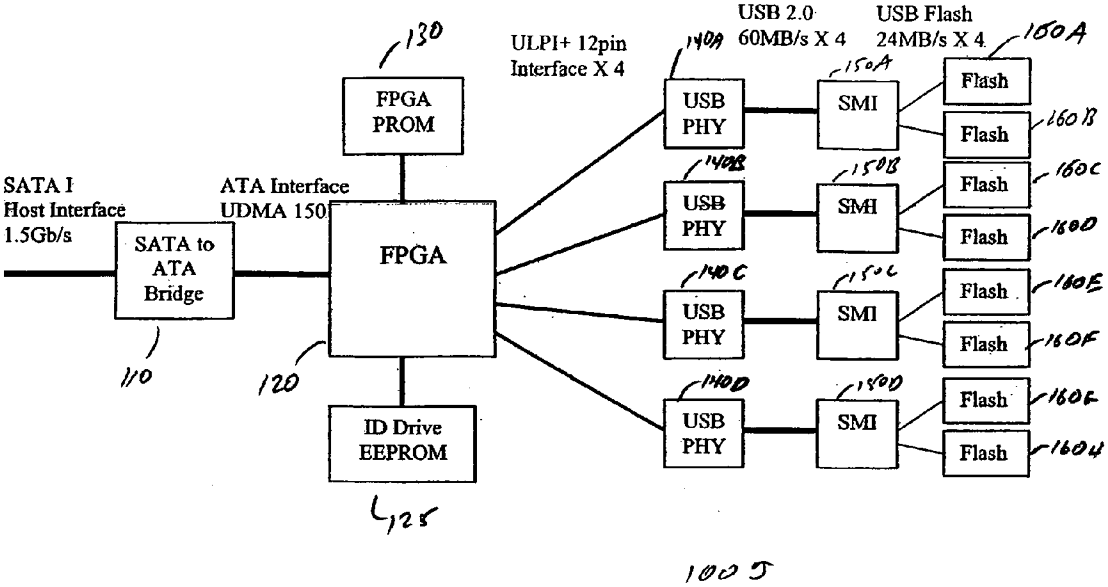

SATA I
Host Interface
1.5Gb/s
SATA to ATA Bridge
110
ATA Interface
UDMA 150
FPGA
120
FPGA PROM
130
ID Drive EEPROM
125
ULPI+ 12pin
Interface X 4
USB 2.0
60MB/s X 4
USB Flash
24MB/s X 4
USB PHY
140A
140B
140C
140D
SMI
150A
150B
150C
150D
Flash
160A
160B
160C
160D
160E
160F
160G
160H
100

Fig. 1

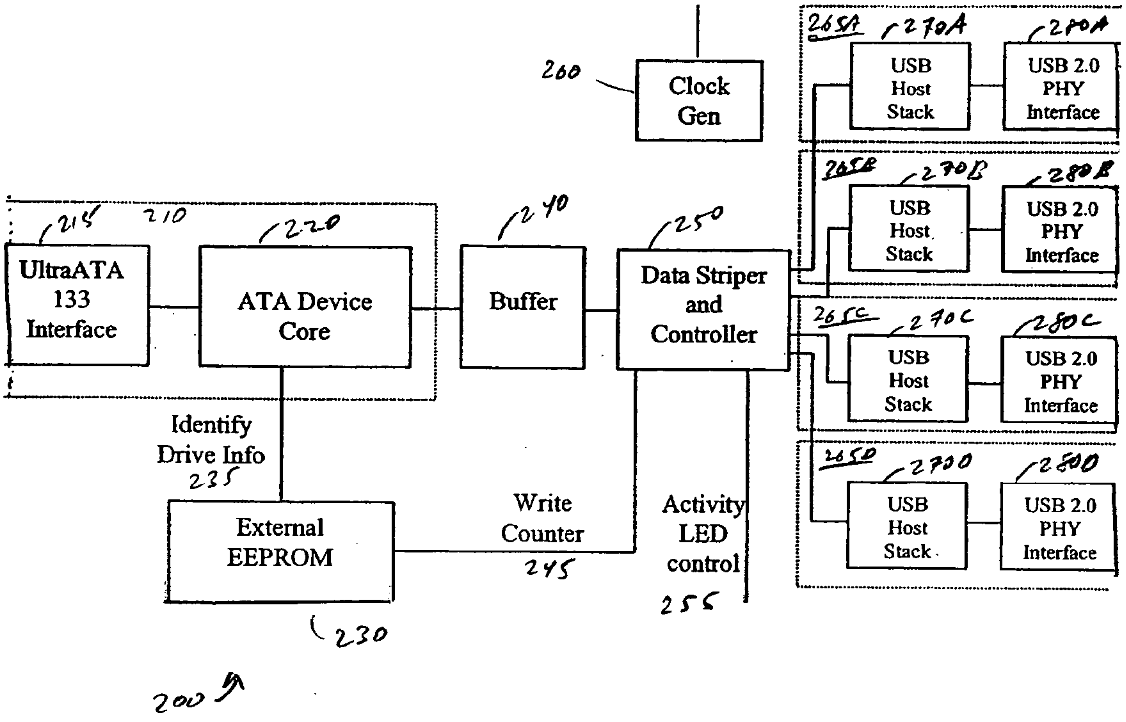

260
Clock Gen
265A
270A
280A
USB Host Stack
USB 2.0 PHY Interface
265B
270B
280B
USB Host Stack
USB 2.0 PHY Interface
215
210
220
240
250
UltraATA 133 Interface
ATA Device Core
Buffer
Data Striper and Controller
265C
270C
280C
USB Host Stack
USB 2.0 PHY Interface
Identify Drive Info
235
265D
270D
280D
USB Host Stack
USB 2.0 PHY Interface
External EEPROM
Write Counter
245
Activity LED control
255
230
200

Fig. 2

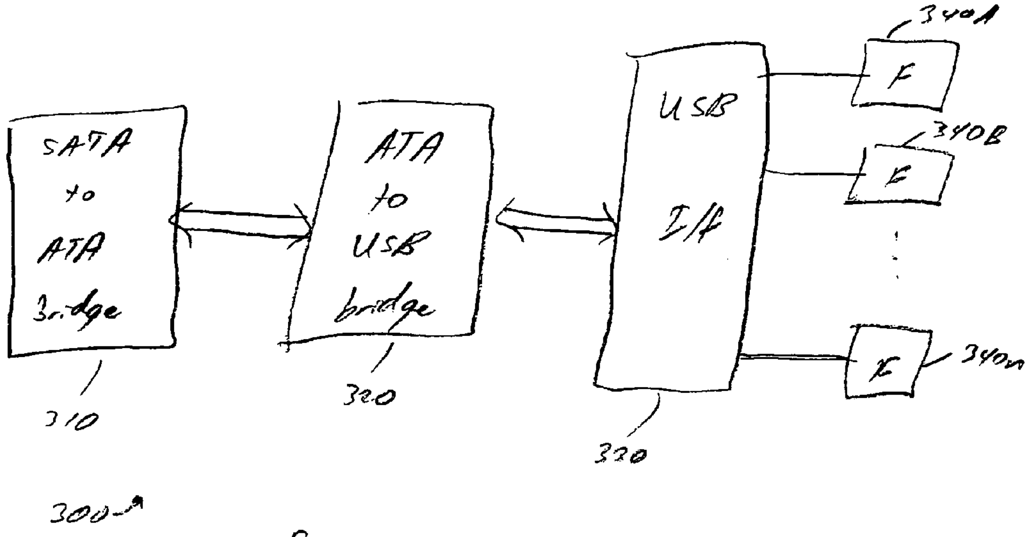

SATA to ATA Bridge
ATA to USB bridge
USB I/F
F
F
F
340A
340B
340n
310
320
330
300

Fig. 3

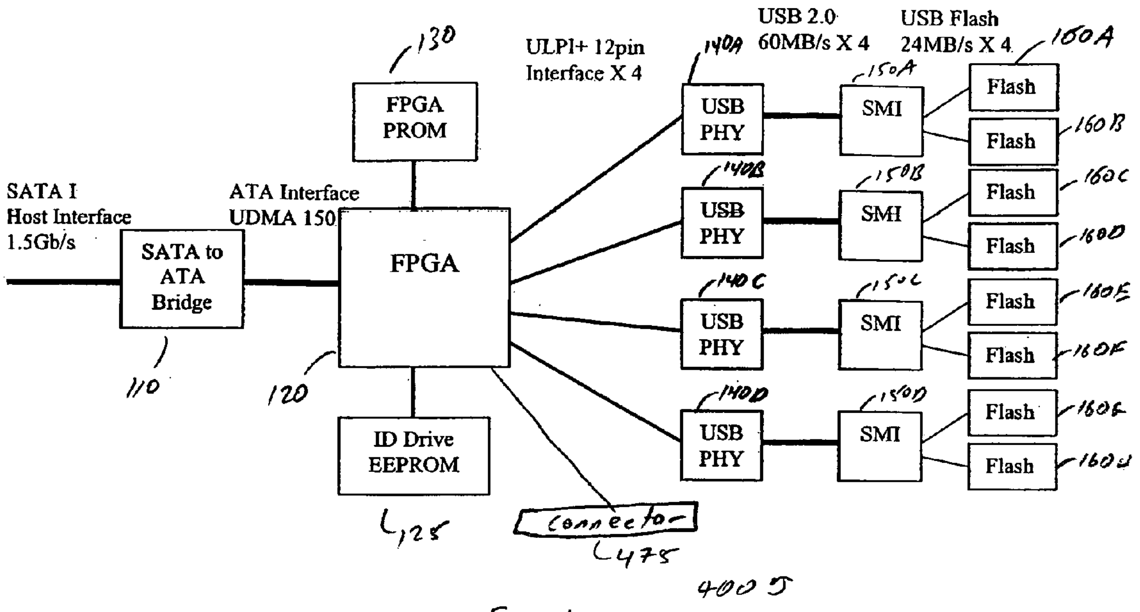

SATA I
Host Interface
1.5Gb/s
SATA to ATA Bridge
110
ATA Interface
UDMA 150
FPGA
120
FPGA PROM
130
ID Drive EEPROM
125
Connector
475
ULPI+ 12pin
Interface X 4
USB 2.0
60MB/s X 4
USB Flash
24MB/s X 4
USB PHY
140A
140B
140C
140D
SMI
150A
150B
150C
150D
Flash
160A
160B
160C
160D
160E
160F
160G
160H
400

Fig. 4

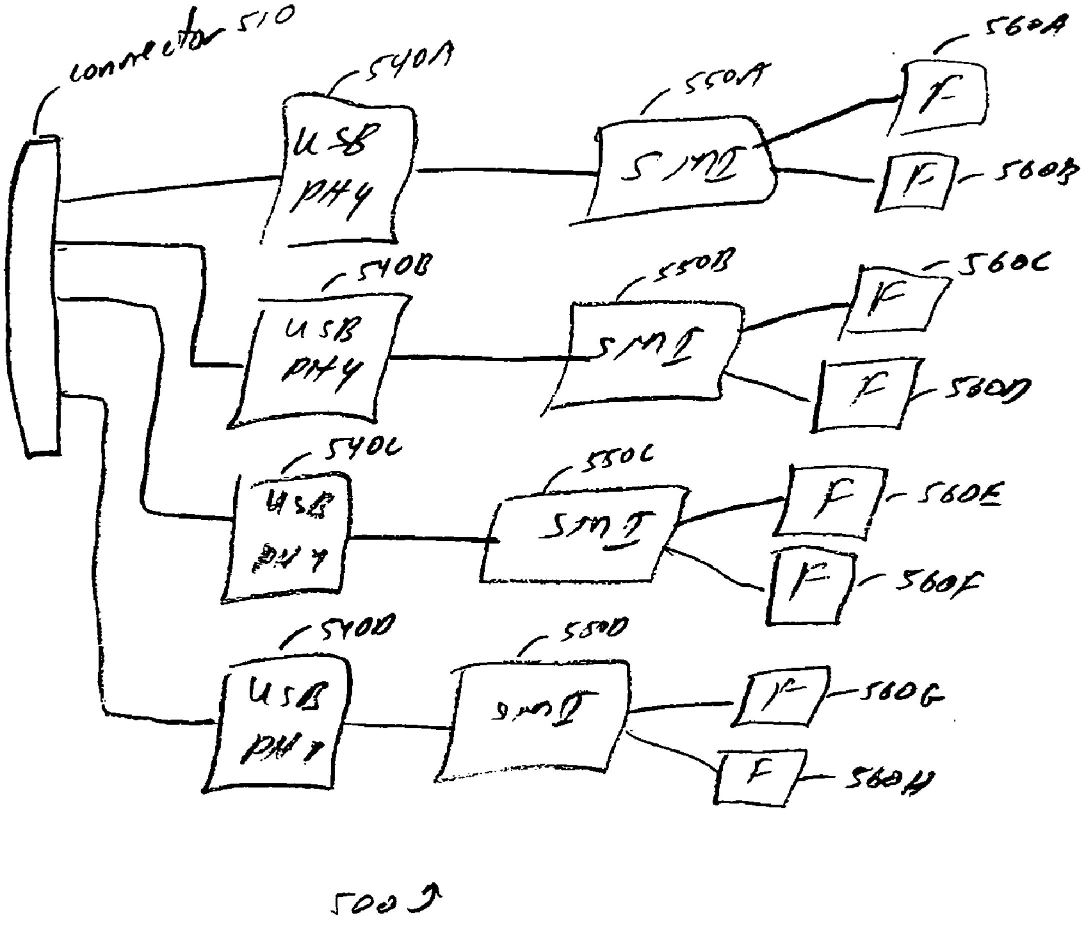

connector 510
540A
USB PHY
540B
USB PHY
540C
USB PHY
540D
USB PHY
550A
SMI
550B
SMI
550C
SMI
550D
SMI
560A
F
560B
F
560C
F
560D
F
560E
F
560F
F
560G
F
560H
F
500

Fig. 5

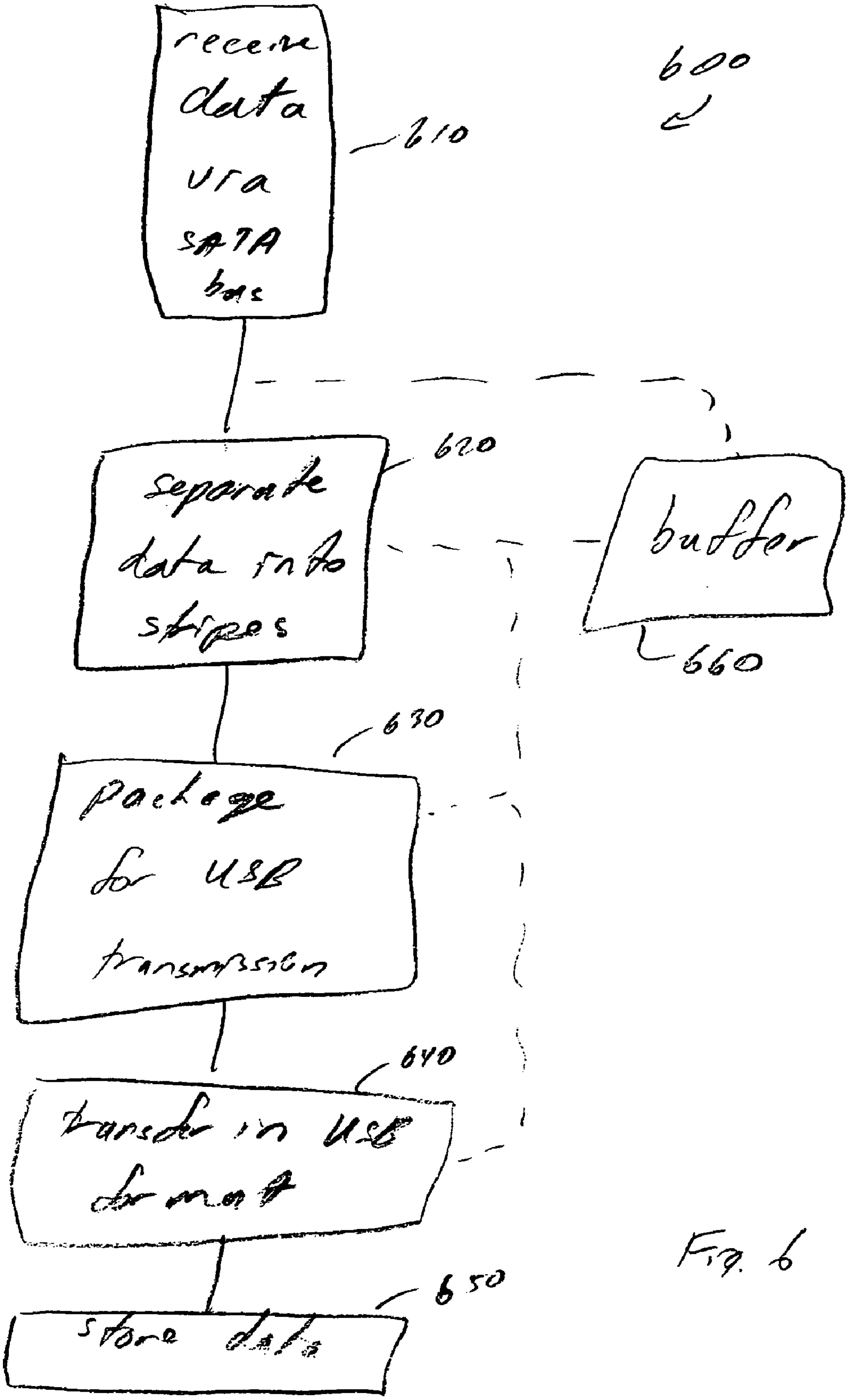

600
receive data via SATA bus
610
separate data into stripes
620
buffer
660
package for USB transmission
630
transfer in USB format
640
store data
650

Fig. 6

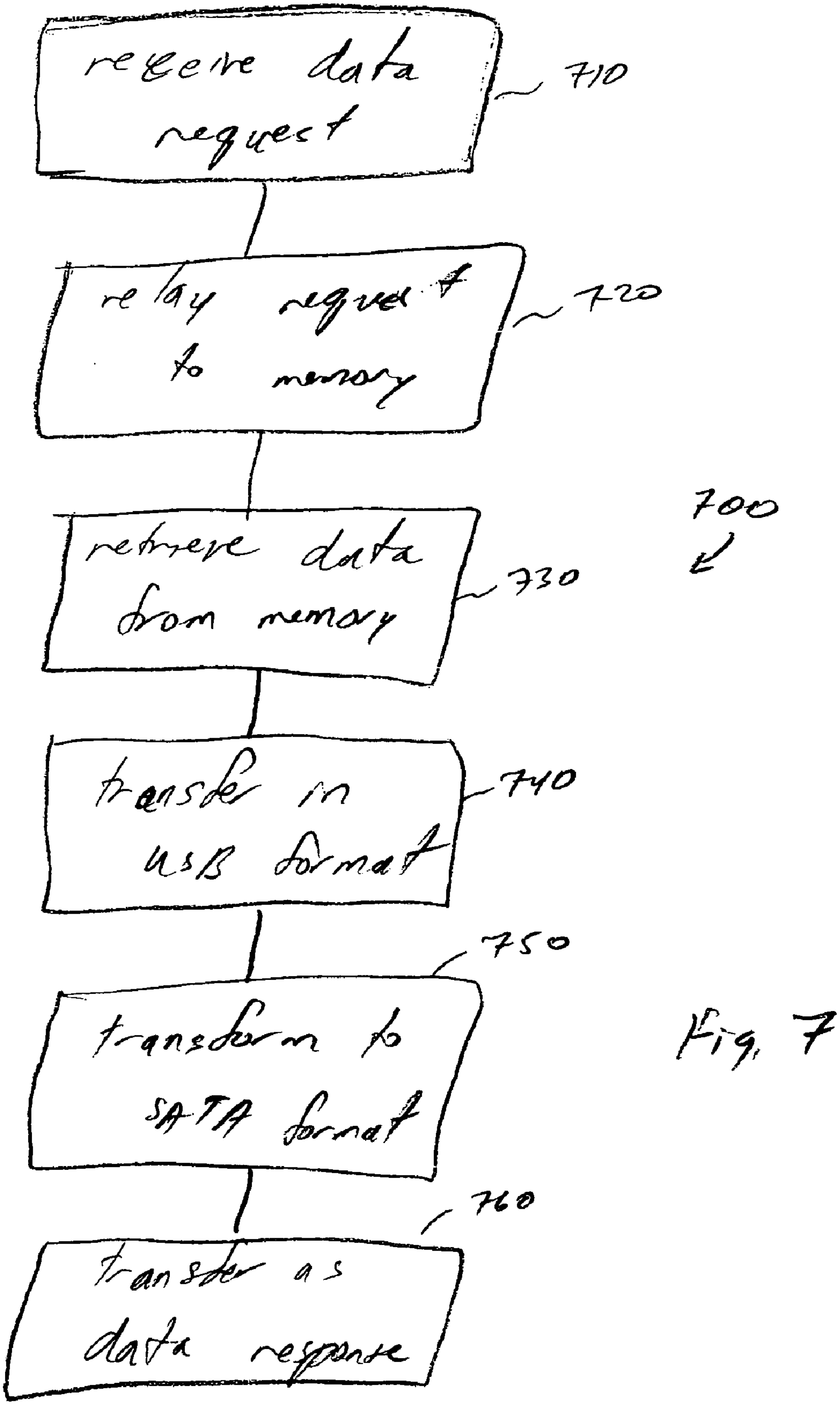
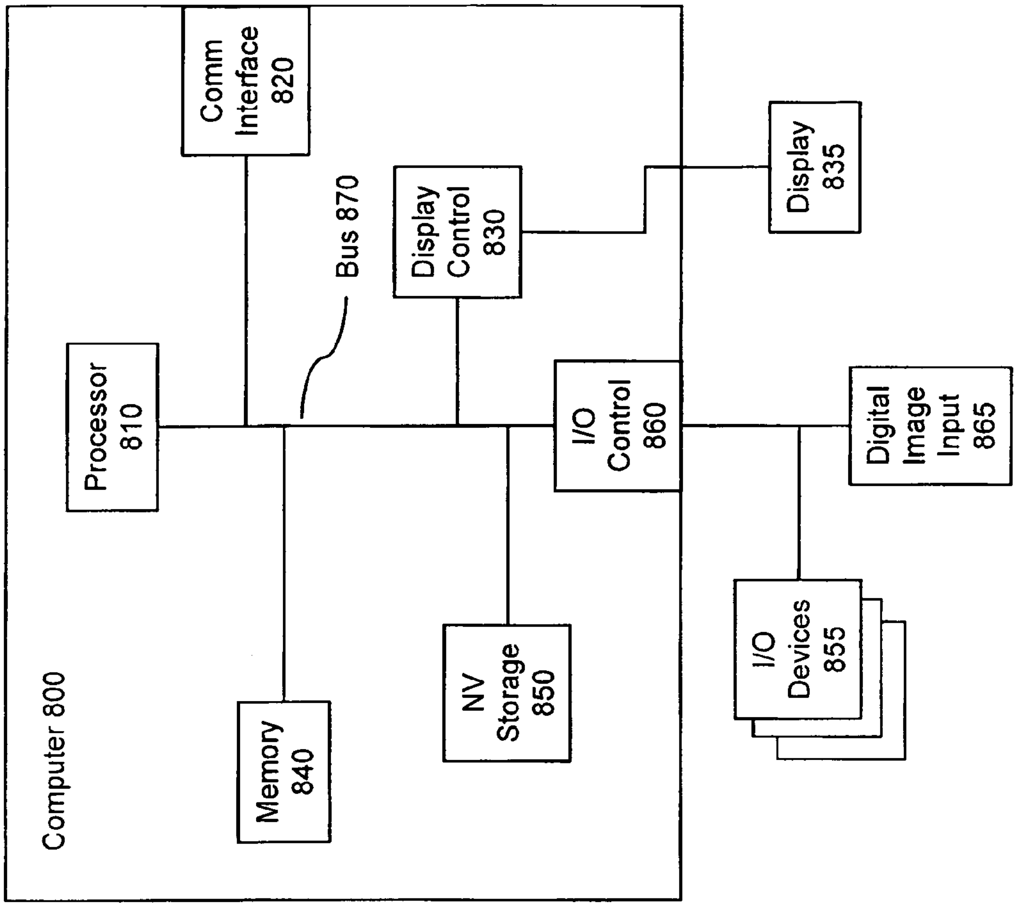

receive data request
710
relay request to memory
720
700
retrieve data from memory
730
transfer in USB format
740
750
transform to SATA format
760
transfer as data response

Fig. 7

FIG. 8


Computer 800
Processor 810
Comm Interface 820
Display Control 830
Display 835
Memory 840
NV Storage 850
I/O Devices 855
I/O Control 860
Digital Image Input 865
Bus 870

DATA STRIPING TO FLASH MEMORY

BACKGROUND

[0001] ATA and Serial ATA (SATA) interfaces are well known for use with disk drives and similar mass storage devices. This wide knowledge makes these interfaces popular among system designers and those who approve system designs for reliability and quality. The identifiable name or acronym provides a basic level of comfort to everyone involved, along with the requirements that related standards impose on devices which are said to be ATA or SATA compatible.

[0002] In many embedded systems, a hard drive would be overkill in some ways and budget-busting in other ways. Additionally, the hard drive might not be reliable under the conditions in which the embedded system may operate. Excess vibration, for example, may ruin a hard drive in short order. Similarly, while a 100 MB hard drive was a large storage volume in the early 1990s, it is a small storage volume today. Thus, embedding a hard drive as part of an embedded system may be unwise.

[0003] One may then attempt to provide an alternative to a hard drive in an embedded system. One may desire that the alternative provide reliability without all of the moving parts of a hard drive, for example. Likewise, one may desire that the alternative provide an interface which is very well understood and characterized. Thus, it may be useful to provide an embedded hard drive with a SATA interface and a reliable structure, but without significant moving parts.

[0004] Moreover, one may desire an expandable format. In some systems, the ability to add more storage capacity at a later date may be very valuable. Thus, it may be useful to provide a drive design that allows for expansion.

BRIEF DESCRIPTION OF THE DRAWINGS

[0005] The present invention is illustrated by way of example in the accompanying drawings. The drawings should be understood as illustrative rather than limiting.

[0006] FIG. **1** illustrates an embodiment of a system for implementing a striped FLASH drive with a SATA interface.

[0007] FIG. **2** illustrates an embodiment of a controller for use in the system of FIG. **1**.

[0008] FIG. **3** illustrates an alternate embodiment of a system for implementing a striped FLASH drive with a SATA interface.

[0009] FIG. **4** illustrates an alternate embodiment of a system for implementing a striped FLASH drive with a SATA interface.

[0010] FIG. **5** illustrates an embodiment of a striped FLASH drive with a USB interface.

[0011] FIG. **6** illustrates an embodiment of a method of writing to a striped FLASH drive with a SATA interface.

[0012] FIG. **7** illustrates an embodiment of a method of reading from a striped FLASH drive with a SATA interface.

[0013] FIG. **8** illustrates an embodiment of a system in which a striped FLASH drive with a SATA interface may be used.

DETAILED DESCRIPTION

[0014] A system, method and apparatus is provided for data striping to FLASH memory. The specific embodiments described in this document represent examples or embodiments of the present invention, and are illustrative in nature rather than restrictive. Implementing a USB-based FLASH drive with a Serial ATA (SATA) interface may be achieved in a variety of ways, and may bring a number of benefits. An interface which translates SATA to ATA signals and then ATA to USB signals may be used to allow for use of relatively common and inexpensive USB FLASH memory in a FLASH drive. This potentially allows for an effective product for use under constraints of either cost or physical reliability.

[0015] In one embodiment, an apparatus is provided. The apparatus includes an SATA to ATA bridge, an ATA to USB bridge coupled to the SATA to ATA bridge, and a USB interface coupled to the ATA to USB bridge. The apparatus also includes a first FLASH memory controller coupled to the USB interface. The apparatus further includes a first FLASH memory module coupled to the first FLASH memory controller. The apparatus also includes a second FLASH memory controller coupled to the USB interface and a second FLASH memory module coupled to the second FLASH memory controller.

[0016] In another embodiment, a method is provided. The method includes receiving data via an SATA bus. The method further includes translating the data into ATA format. The method also includes separating the data into stripes. Moreover, the method includes packaging the data for USB transmission. Additionally, the method includes transferring the data in USB format. Also, the method includes storing the data in a set of FLASH memory modules.

[0017] In yet another embodiment, a method is provided. The method includes receiving a data request in SATA format. Additionally, the method includes translating the request into USB format. Moreover, the method includes relaying the request to memory. Likewise, the method includes retrieving corresponding data from memory. Also, the method includes transferring the corresponding data in USB format and transforming the corresponding data to SATA format. Furthermore, the method includes transferring the corresponding data as a response to the request.

[0018] In the following description, for purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of the invention. It will be apparent, however, to one skilled in the art that the invention can be practiced without these specific details. In other instances, structures and devices are shown in block diagram form in order to avoid obscuring the invention.

[0019] Reference in the specification to "one embodiment" or "an embodiment" means that a particular feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment of the invention. The appearances of the phrase "in one embodiment" in various places in the specification are not necessarily all referring to the same embodiment, nor are separate or alternative embodiments mutually exclusive of other embodiments. Features and aspects of various embodiments maybe integrated into other embodiments, and embodiments illustrated in this document may be implemented without all of the features or aspects illustrated or described.

[0020] Various embodiments of a SATA interface for a FLASH drive may be implemented. FIG. **1** illustrates an embodiment of a system for implementing a striped FLASH drive with a SATA interface. System **100** includes an SATA to ATA bridge, an FPGA serving as an ATA to USB bridge, USB FLASH modules including a physical layer, controller and FLASH chip, and associated ROMs. Similarly implementations may be envisioned, using ASICs or other types of FPGAs, or using other USB FLASH controllers, for example.

[0021] A system using drive **100** (such as for a write) initially sees the SATA to ATA bridge **110**—so the drive **100** looks like any other SATA drive to an external system. FPGA **120** is coupled to bridge **110** to receive ATA protocol signals. In one embodiment, FPGA **120** is a Xilinx RAM-based FPGA, with configuration data stored in FPGA PROM **130** and drive identification data stored in ID EEPROM **125**. Other types of FPGAs, such as EEPROM FPGAs may similarly be used.

[0022] FPGA **120** provides a bridge from ATA protocol signals to USB protocol signals, and routes USB signals to USB physical module **140**A-D. Each USB physical module **140** may receive data for one of four SMI FLASH controllers **150**A-D, and thereby for one of eight FLASH modules **160**A-H. Thus, FPGA **120** may not only translate signals to a USB format, but also stripe data—sending bits to proper FLASH module **160**A-H. Moreover, RAM available in FPGA **120** may be used to provide a buffer for transactions where throughput does not meet the quantity of data to be written. SMI FLASH controllers **150**A-D are controllers for FLASH memory modules, which handle the specifics of driving address and data rows (or reading data rows) and providing a simple interface from which data may be retrieved. Other FLASH memory controllers may be similarly useful.

[0023] For a read operation, an external system still sees an SATA interface at bridge **110**. However, rather than steering data to proper FLASH modules **160**A-H, an address is presented at bridge **110** and propagated through (potentially with some translations) until a local address is presented to each of FLASH modules **160**A-H to retrieve a corresponding bit. Thus, the data is striped—an MSB of each datum may be stored in module **160**A with the corresponding LSB stored in module **160**H, for example. Note that in some embodiments, striping may involve a different splitting of data.

[0024] Reference to the controller in a system such as drive **100** may be helpful for understanding. FIG. **2** illustrates an embodiment of a controller for use in the system of FIG. **1**. System **200** shows the controller and some of the interface circuitry of surrounding components in some embodiments. Interface **210** provides an SATA to ATA bridge, using UltraATA interface **215** and ATA device core **220**. Physical modules **265**A-D provide USB physical modules for various FLASH modules, with a USB host stack **270**A-D and USB physical interface **280**A-D both included. The controller operates between these two interfaces, as the glue logic that allows data exchange between the two interfaces.

[0025] Buffer **240** buffers incoming and outgoing requests from the SATA interface **210**. Data striper and controller **250** determines where data should go, or where data should be requested from among the various channels for data in the FLASH modules. Thus, data striper **250** maintains information about which bits of a data byte or word go to each module (through each physical module **265**), for example. Controller **250** also operates an activity LED control **255** (which may be a simple signal, for example). Controller **250** also controls write counter signal **245** and thereby EEPROM **230**, which may be used for reliability purposes, for example. EEPROM **230** provides drive identifying information **235**, and may also include reliability information in some embodiments, for example. Also included in the circuit is clock generator **260**, which is used for synchronous operations when needed.

[0026] FIG. **3** illustrates an alternate embodiment of a system for implementing a striped FLASH drive with a SATA interface. This provides an overview of the general system implemented in various embodiments in FIGS. **1** and **2**. System **300** includes an SATA to ATA bridge **310** coupled to an ATA to USB bridge **320**, further coupled to a USB interface **330**, which communicates with a series of FLASH modules **340**A-n. Such a system may thus translate SATA protocol to ATA protocol, and then ATA protocol to USB protocol for purposes of storing and retrieving data. Along with protocol translations, address translations may be involved, as system addresses may differ from drive-level local addresses, for example. Moreover, buffering in one or more bridges may be involved due to differences in how the various protocols handle processing delays.

[0027] As is apparent from reviewing FIG. **3**, more than eight FLASH modules may be used in a single drive. This can be handled with more modules incorporated in a single board, for example. FIG. **4** illustrates an alternate embodiment of a system for implementing a striped FLASH drive with a SATA interface. Board **400** is a FLASH drive with a SATA interface such as was described with respect to FIG. **1**. However, connector **475** is also provided. Connector **475** allows for connection of a daughter board. Thus, if a form factor for board **400** prevents more than eight FLASH modules **160** being included, a daughter board with more FLASH modules may be connected.

[0028] FIG. **5** illustrates an embodiment of a striped FLASH drive with a USB interface. Board **500** provides a daughter board in some embodiments, which can be interfaced or connected with board **400** to provide more data storage. Connector **510** provides a connection through direct connection or through a cable to a connector such as connector **475**. Coupled to connector **510** are USB physical modules **540**A-D. Coupled to USB physical modules **540**A-D are FLASH controllers **550**A-D, and coupled thereto are FLASH modules **560**A-G. Thus, a second set of eight FLASH modules may be provided, in one embodiment, to the drive **400** of FIG. **4**. This may allow for striping of sixteen bits, or for additional memory locally addressed in a logically different space from the FLASH memory of drive **400**, for example. Also, note that a daughter board may have multiple connectors, allowing for further expansion (such as by cable, for example). Similarly, a main or mother board may have multiple connectors or slots for expansion, too.

[0029] Processes used by SATA FLASH drives can vary a fair amount, but the following basic process may be useful in some embodiments. FIG. **6** illustrates an embodiment of a method of writing to a striped FLASH drive with a SATA interface. Process **600** includes receiving data via an SATA bus, separating the data into stripes, buffering if necessary, packaging for USB transmission, transferring the data in USB format, and storing the data. Process **600** and other processes of this document are implemented as a set of modules, which may be process modules or operations, software modules with associated functions or effects, hardware modules designed to fulfill the process operations, or some combination of the various types of modules, for example. The modules of process **600** and other processes described herein may be rearranged, such as in a parallel or serial fashion, and may be reordered, combined, or subdivided in various embodiments.

[0030] Process **600** initiates with receipt of data via an SATA bus at module **610**. At module **620**, the data is separated into stripes. The data may also be buffered at module **660**, whether at the time of receipt or later in the process. The data is also transformed from SATA format to ATA format either before or after striping at module **620**. At module **630**, the data is packaged for USB transmission (e.g. it is transformed from ATA format to USB format). At module **640**, the data is actually transferred in USB format, and at module **650**, the data is stored in FLASH memory modules.

[0031] Just as data may be stored, it may be retrieved, through various different processes. FIG. **7** illustrates an embodiment of a method of reading from a striped FLASH drive with a SATA interface. Process **700** includes receiving a data request, relaying the request to memory, retrieving data from memory, transferring the data in USB format, transforming the data to SATA format, and transferring the data as a response.

[0032] Process **700** initiates at module **710** with receipt of a data request. The data request is relayed to memory at module **720**. This may involve translation from SATA format through ATA format into USB format, with address translations as well. The data is then retrieved from memory at module **730**. At module **740**, the data is transferred in USB format. At module **750**, the data is transformed to SATA format—this may include an intermediate transformation to ATA format. At module **760**, the retrieved data is provided to an external system as a response to the request of module **710**.

[0033] FIG. **8** illustrates an embodiment of a system in which a striped FLASH drive with a SATA interface may be used. The following description of FIG. **8** is intended to provide an overview of device hardware and other operating components suitable for performing the methods of the invention described above and hereafter, but is not intended to limit the applicable environments. Similarly, the hardware and other operating components may be suitable as part of the apparatuses described above. The invention can be practiced with other system configurations, including personal computers, multiprocessor systems, microprocessor-based or programmable consumer electronics, network PCs, minicomputers, mainframe computers, embedded devices or components, and the like. The invention can also be practiced in distributed computing environments where tasks are performed by remote processing devices that are linked through a communications network.

[0034] FIG. **8** shows one example of a personal device that can be used as a cellular telephone or similar personal device, or may be used as a more conventional personal computer, as an embedded processor or local console, or as a PDA, for example. Such a device can be used to perform many functions depending on implementation, such as monitoring functions, user interface functions, telephone communications, two-way pager communications, personal organizing, or similar functions. The system **800** of FIG. **8** may also be used to implement other devices such as a personal computer, network computer, or other similar systems. The computer system **800** interfaces to external systems through the communications interface **820**. In a cellular telephone, this interface is typically a radio interface for communication with a cellular network, and may also include some form of cabled interface for use with an immediately available personal computer. In a two-way pager, the communications interface **820** is typically a radio interface for communication with a data transmission network, but may similarly include a cabled or cradled interface as well. In a personal digital assistant, communications interface **820** typically includes a cradled or cabled interface, and may also include some form of radio interface such as a Bluetooth or 802.11 interface, or a cellular radio interface for example.

[0035] The computer system **800** includes a processor **810**, which can be a conventional microprocessor such as an Intel pentium microprocessor or Motorola power PC microprocessor, a Texas Instruments digital signal processor, or some combination of the various types or processors. Memory **840** is coupled to the processor **810** by a bus **870**. Memory **840** can be dynamic random access memory (dram) and can also include static ram (sram), or may include FLASH EEPROM, too. The bus **870** couples the processor **810** to the memory **840**, also to non-volatile storage **850**, to display controller **830**, and to the input/output (I/O) controller **860**. Note that the display controller **830** and I/O controller **860** may be integrated together, and the display may also provide input.

[0036] The display controller **830** controls in the conventional manner a display on a display device **835** which typically is a liquid crystal display (LCD) or similar flat-panel, small form factor display. The input/output devices **855** can include a keyboard, or stylus and touch-screen, and may sometimes be extended to include disk drives, printers, a scanner, and other input and output devices, including a mouse or other pointing device. The display controller **830** and the I/O controller **860** can be implemented with conventional well known technology. A digital image input device **865** can be a digital camera which is coupled to an I/O controller **860** in order to allow images from the digital camera to be input into the device **800**.

[0037] The non-volatile storage **850** is often a FLASH memory or read-only memory, or some combination of the two. A magnetic hard disk, an optical disk, or another form of storage for large amounts of data may also be used in some embodiments, though the form factors for such devices typically preclude installation as a permanent component of the device **800**. Rather, a mass storage device on another computer is typically used in conjunction with the more limited storage of the device **800**. Some of this data is often written, by a direct memory access process, into memory **840** during execution of software in the device **800**. One of skill in the art will immediately recognize that the terms "machine-readable medium" or "computer-readable medium" includes any type of storage device that is accessible by the processor **810** and also encompasses a carrier wave that encodes a data signal.

[0038] The device **800** is one example of many possible devices which have different architectures. For example, devices based on an Intel microprocessor often have multiple buses, one of which can be an input/output (**1**/O) bus for the peripherals and one that directly connects the processor **810** and the memory **840** (often referred to as a memory bus). The buses are connected together through bridge components that perform any necessary translation due to differing bus protocols.

[0039] In addition, the device **800** is controlled by operating system software which includes a file management system, such as a disk operating system, which is part of the operating system software. One example of an operating system software with its associated file management system software is the family of operating systems known as Windows CE® and Windows® from Microsoft Corporation of Redmond, Wash., and their associated file management systems. Another example of an operating system software with its associated file management system software is the Palm® operating system and its associated file management system.

The file management system is typically stored in the non-volatile storage **850** and causes the processor **810** to execute the various acts required by the operating system to input and output data and to store data in memory, including storing files on the non-volatile storage **850**. Other operating systems may be provided by makers of devices, and those operating systems typically will have device-specific features which are not part of similar operating systems on similar devices. Similarly, WinCE® or Palm® operating systems may be adapted to specific devices for specific device capabilities.

[0040] Device **800** may be integrated onto a single chip or set of chips in some embodiments, and typically is fitted into a small form factor for use as a personal device. Thus, it is not uncommon for a processor, bus, onboard memory, and display/I-O controllers to all be integrated onto a single chip. Alternatively, functions may be split into several chips with point-to-point interconnection, causing the bus to be logically apparent but not physically obvious from inspection of either the actual device or related schematics.

[0041] Some portions of the detailed description are presented in terms of algorithms and symbolic representations of operations on data bits within a computer memory. These algorithmic descriptions and representations are the means used by those skilled in the data processing arts to most effectively convey the substance of their work to others skilled in the art. An algorithm is here, and generally, conceived to be a self-consistent sequence of operations leading to a desired result. The operations are those requiring physical manipulations of physical quantities. Usually, though not necessarily, these quantities take the form of electrical or magnetic signals capable of being stored, transferred, combined, compared, and otherwise manipulated. It has proven convenient at times, principally for reasons of common usage, to refer to these signals as bits, values, elements, symbols, characters, terms, numbers, or the like.

[0042] It should be borne in mind, however, that all of these and similar terms are to be associated with the appropriate physical quantities and are merely convenient labels applied to these quantities. Unless specifically stated otherwise as apparent from the following discussion, it is appreciated that throughout the description, discussions utilizing terms such as “processing” or “computing” or “calculating” or “determining” or “displaying” or the like, refer to the action and processes of a computer system, or similar electronic computing device, that manipulates and transforms data represented as physical (electronic) quantities within the computer system’s registers and memories into other data similarly represented as physical quantities within the computer system memories or registers or other such information storage, transmission or display devices.

[0043] The present invention, in some embodiments, also relates to apparatus for performing the operations herein. This apparatus may be specially constructed for the required purposes, or it may comprise a general purpose computer selectively activated or reconfigured by a computer program stored in the computer. Such a computer program may be stored in a computer readable storage medium, such as, but is not limited to, any type of disk including floppy disks, optical disks, CD-ROMs, and magnetic-optical disks, read-only memories (ROMs), random access memories (RAMs), EPROMs, EEPROMs, magnetic or optical cards, or any type of media suitable for storing electronic instructions, and each coupled to a computer system bus.

[0044] The algorithms and displays presented herein are not inherently related to any particular computer or other apparatus. Various general purpose systems may be used with programs in accordance with the teachings herein, or it may prove convenient to construct more specialized apparatus to perform the required method steps. The required structure for a variety of these systems will appear from the description below. In addition, the present invention is not described with reference to any particular programming language, and various embodiments may thus be implemented using a variety of programming languages.

[0045] One skilled in the art will appreciate that although specific examples and embodiments of the system and methods have been described for purposes of illustration, various modifications can be made without deviating from the present invention. For example, embodiments of the present invention may be applied to many different types of databases, systems and application programs. Moreover, features of one embodiment may be incorporated into other embodiments, even where those features are not described together in a single embodiment within the present document.

What is claimed is:

1. An apparatus, comprising:

an SATA to ATA bridge;

an ATA to USB bridge coupled to the SATA to ATA bridge;

a USB interface coupled to the ATA to USB bridge;

a first FLASH memory controller coupled to the USB interface;

a first FLASH memory module coupled to the first FLASH memory controller;

a second FLASH memory controller coupled to the USB interface;

and

a second FLASH memory module coupled to the second FLASH memory controller.

2. The apparatus of claim **1**, further comprising:

a buffer coupled to the ATA to USB bridge.

3. The apparatus of claim **2**, further comprising:

a third FLASH memory controller coupled to the USB interface;

and

a third FLASH memory module coupled to the third FLASH memory controller.

4. The apparatus of claim **3**, further comprising:

a fourth FLASH memory controller coupled to the USB interface;

and

a fourth FLASH memory module coupled to the fourth FLASH memory controller.

5. The apparatus of claim **4**, further comprising:

a fifth FLASH memory module coupled to the first FLASH memory controller;

a sixth FLASH memory module coupled to the second FLASH memory controller;

a seventh FLASH memory module coupled to the third FLASH memory controller;

and

an eighth FLASH memory module coupled to the fourth FLASH memory controller.

6. The apparatus of claim **5**, wherein:

the ATA to USB interface and the buffer are implemented as an FPGA.

7. The apparatus of claim **6**, further comprising:

a drive identifier EEPROM coupled to the FPGA.

8. The apparatus of claim **6**, further comprising:
a connector coupled to the FPGA.

9. The apparatus of claim **8**, further comprising:
a separate daughter board, the daughter board including:
a mating connector coupled to the connector;
a fifth FLASH memory controller coupled to the mating connector;
a ninth FLASH memory module coupled to the fifth FLASH memory controller;
a sixth FLASH memory controller coupled to the mating connector;
and
a tenth FLASH memory module coupled to the sixth FLASH memory controller.

10. The apparatus of claim **9**, wherein:
the daughter board further includes:
an eleventh FLASH memory module coupled to the fifth FLASH memory controller;
a twelfth FLASH memory module coupled to the sixth FLASH memory controller;
a seventh FLASH memory controller coupled to the mating connector;
a thirteenth FLASH memory module coupled to the seventh FLASH memory controller;
a fourteenth FLASH memory module coupled to the seventh FLASH memory controller;
an eighth FLASH memory controller coupled to the mating connector;
a fifteenth FLASH memory module coupled to the eighth FLASH memory controller;
and
a sixteenth FLASH memory module coupled to the eighth FLASH memory controller.

11. The apparatus of claim **10**, wherein:
the daughter board further includes:
a USB interface coupled to the mating connector, the USB interface coupled between the mating connector and the fifth FLASH memory controller, the sixth FLASH memory controller, the seventh FLASH memory controller and the eighth FLASH memory controller.

12. The apparatus of claim **6**, further comprising:
means for identifying the apparatus electronically.

13. A method, comprising:
receiving data via an SATA bus;
translating the data into ATA format;
separating the data into stripes;
packaging the data for USB transmission;
transferring the data in USB format;
and
storing the data in a set of FLASH memory modules.

14. The method of claim **13**, further comprising:
buffering the data.

15. The method of claim **14**, further comprising:
receiving a data request in SATA format;
translating the request into ATA format;
translating the request into USB format;
relaying the request to the set of FLASH memory modules;
retrieving corresponding data from the set of FLASH memory modules;
transferring the corresponding data in USB format;
transforming the corresponding data to ATA format;
transforming the corresponding data to SATA format;
and
transferring the corresponding data as a response to the request.

16. The method of claim **15**, further comprising:
providing a drive identification responsive to a request.

17. A method, comprising:
receiving a data request in SATA format;
translating the request into USB format;
relaying the request to memory;
retrieving corresponding data from memory;
transferring the corresponding data in USB format;
transforming the corresponding data to SATA format;
and
transferring the corresponding data as a response to the request.

18. The method of claim **17**, further comprising:
translating the request into ATA format;
transforming the corresponding data to ATA format;

19. The method of claim **18**, further comprising:
buffering the data request.

20. The method of claim **17**, further comprising:
providing a drive identification responsive to a request.

* * * * *