US 20060294269A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2006/0294269 A1

Duerk et al. (43) Pub. Date: Dec. 28, 2006

(54) HARD DISK DRIVE STAGGERED SPIN-UP MECHANISM

(76) Inventors: **Vicky Duerk**, Shrewsbury, MA (US); **Pak-lung Seto**, Shrewsbury, MA (US)

Correspondence Address:
BLAKELY SOKOLOFF TAYLOR & ZAFMAN
12400 WILSHIRE BOULEVARD
SEVENTH FLOOR
LOS ANGELES, CA 90025-1030 (US)

(21) Appl. No.: **11/168,782**

(22) Filed: **Jun. 28, 2005**

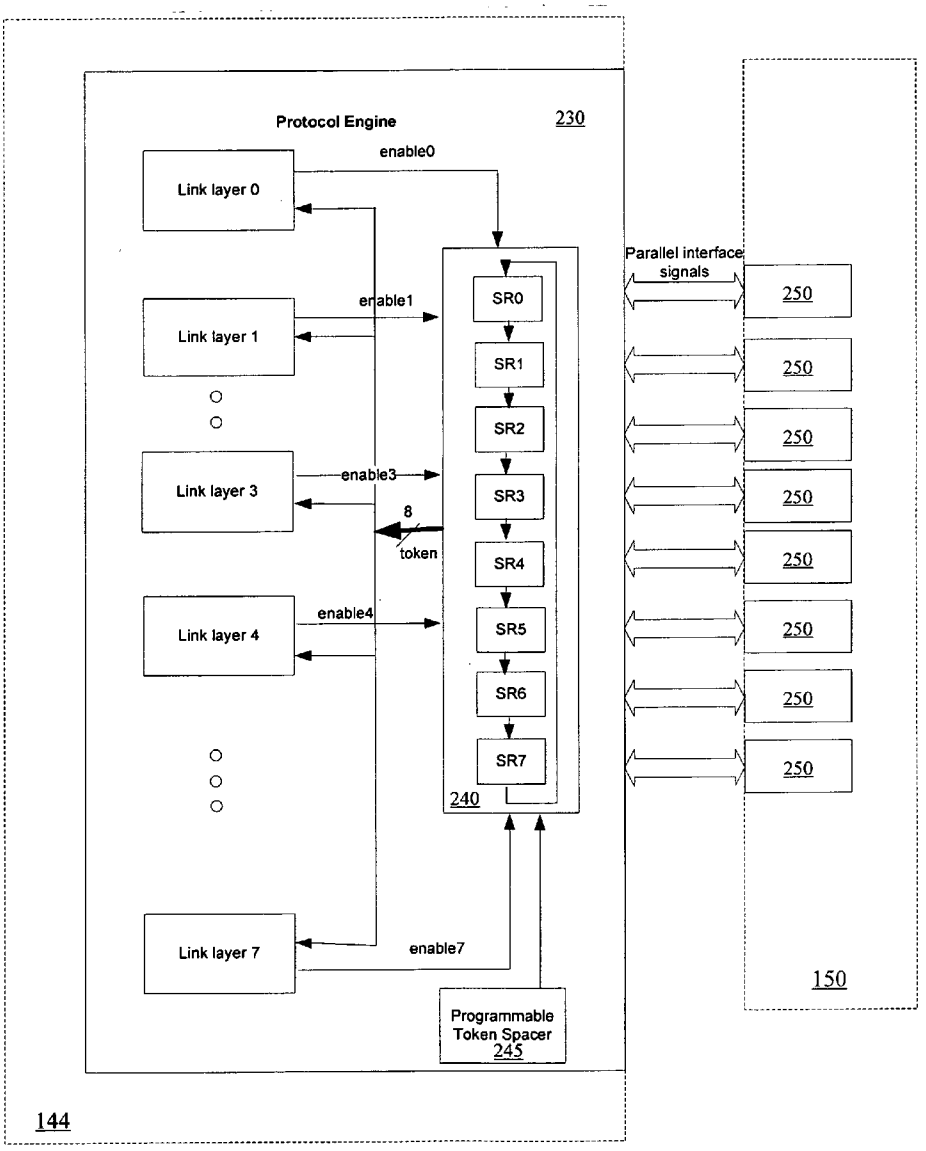
Publication Classification

(51) **Int. Cl.**
G06F 13/38 (2006.01)

(52) **U.S. Cl.** **710/62**

(57) **ABSTRACT**

According to one embodiment, a host bus adapter (HBA) is disclosed. The HBA includes one or more link layer engines, one or more ports, each of the one or more ports associated with one of the one or more link layer engines and token passing logic having a shift register associated with each of the one or more link layer engines. A first link layer engine enables a first storage device coupled to an associated port to spin-up whenever the first link layer engine detects that a first shift register has a first value.



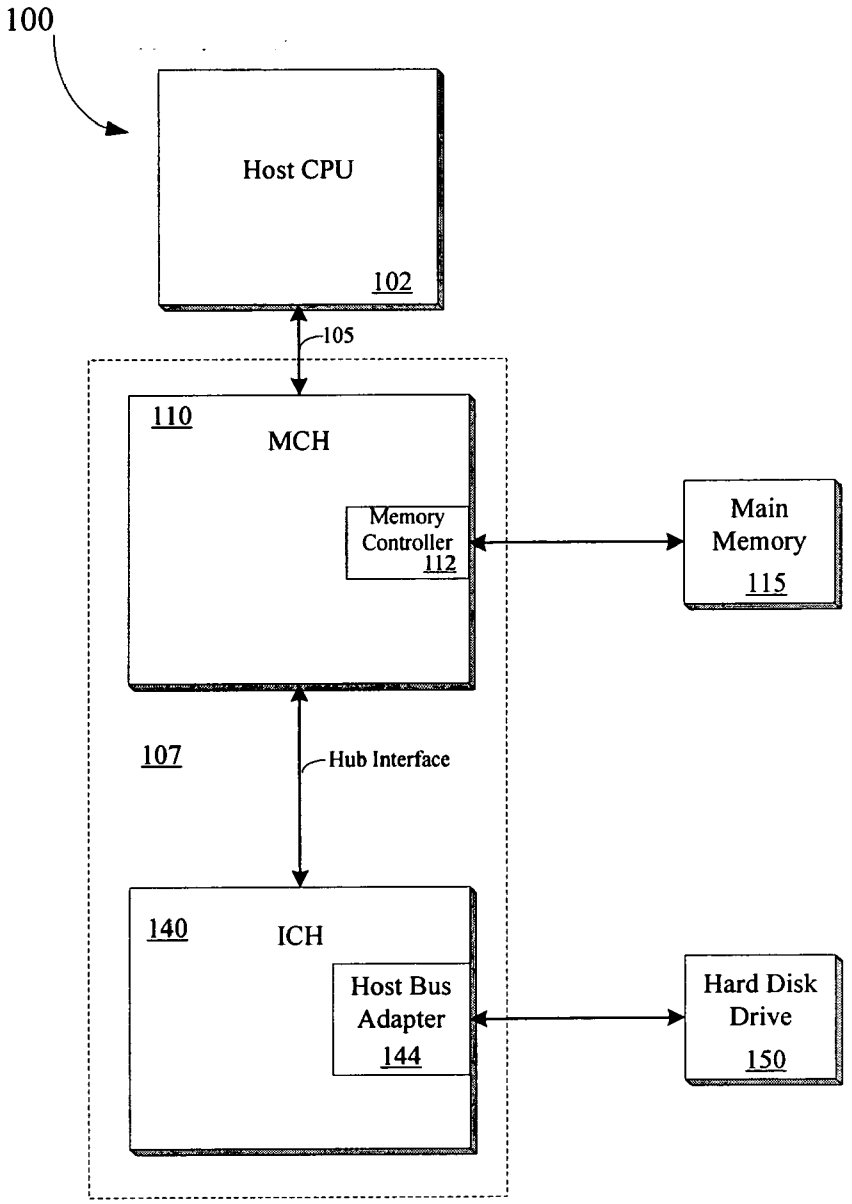

100
Host CPU
102
105
110
MCH
Memory Controller 112
Main Memory 115
107
Hub Interface
140
ICH
Host Bus Adapter 144
Hard Disk Drive 150

Figure 1

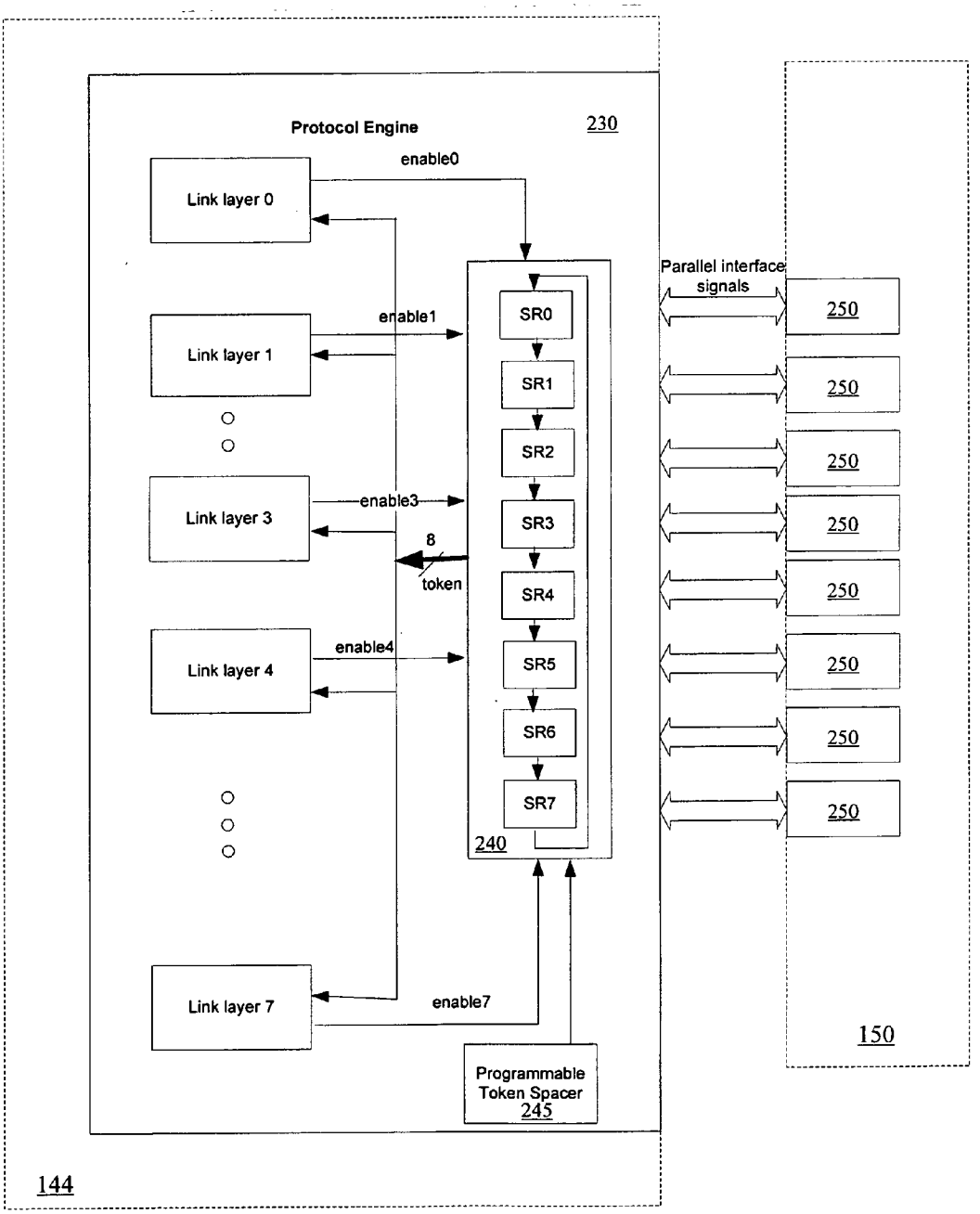

Protocol Engine
230
enable0
Link layer 0
enable1
Link layer 1
Link layer 3
enable3
8
token
Link layer 4
enable4
Link layer 7
enable7
SR0
SR1
SR2
SR3
SR4
SR5
SR6
SR7
240
Programmable
Token Spacer
245
144
Parallel interface
signals
250
250
250
250
250
250
250
150

Figure 2

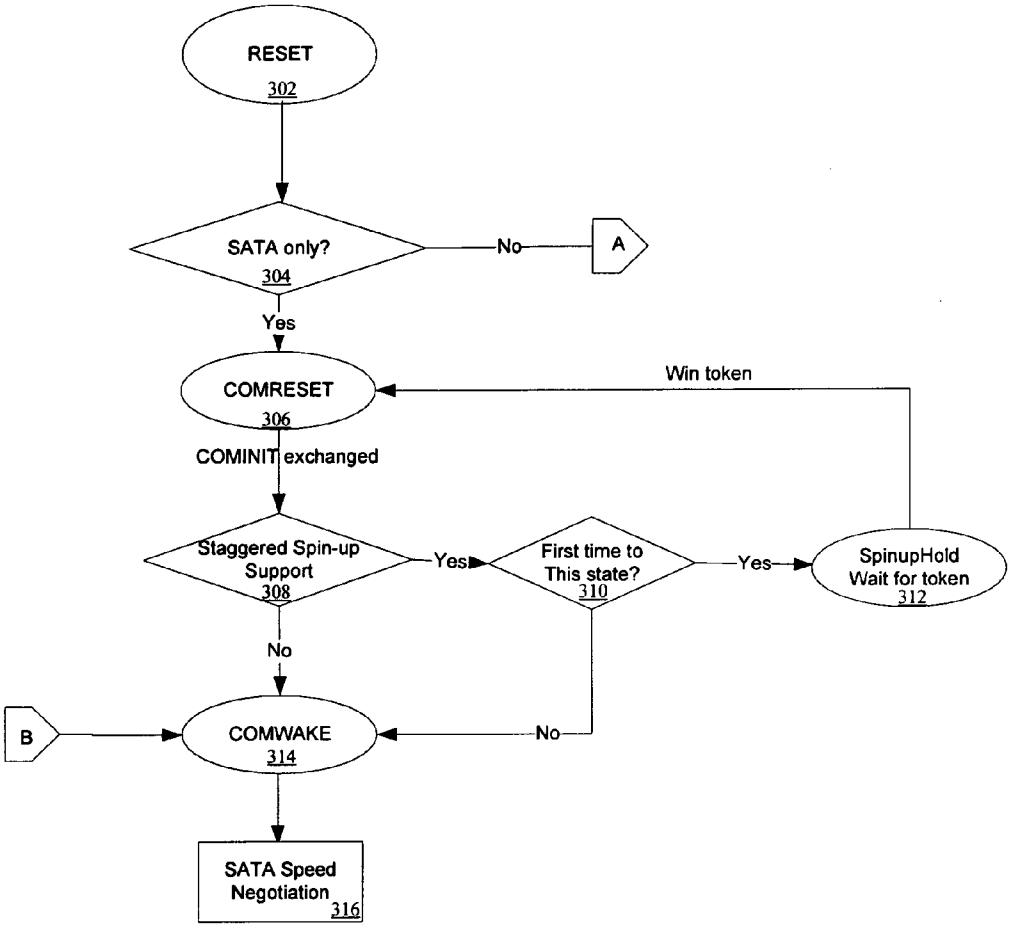

RESET
302
SATA only?
304
No
A
Yes
COMRESET
306
Win token
COMINIT exchanged
Staggered Spin-up Support
308
Yes
First time to This state?
310
Yes
SpinupHold Wait for token
312
No
B
COMWAKE
314
No
SATA Speed Negotiation
316

Figure 3A

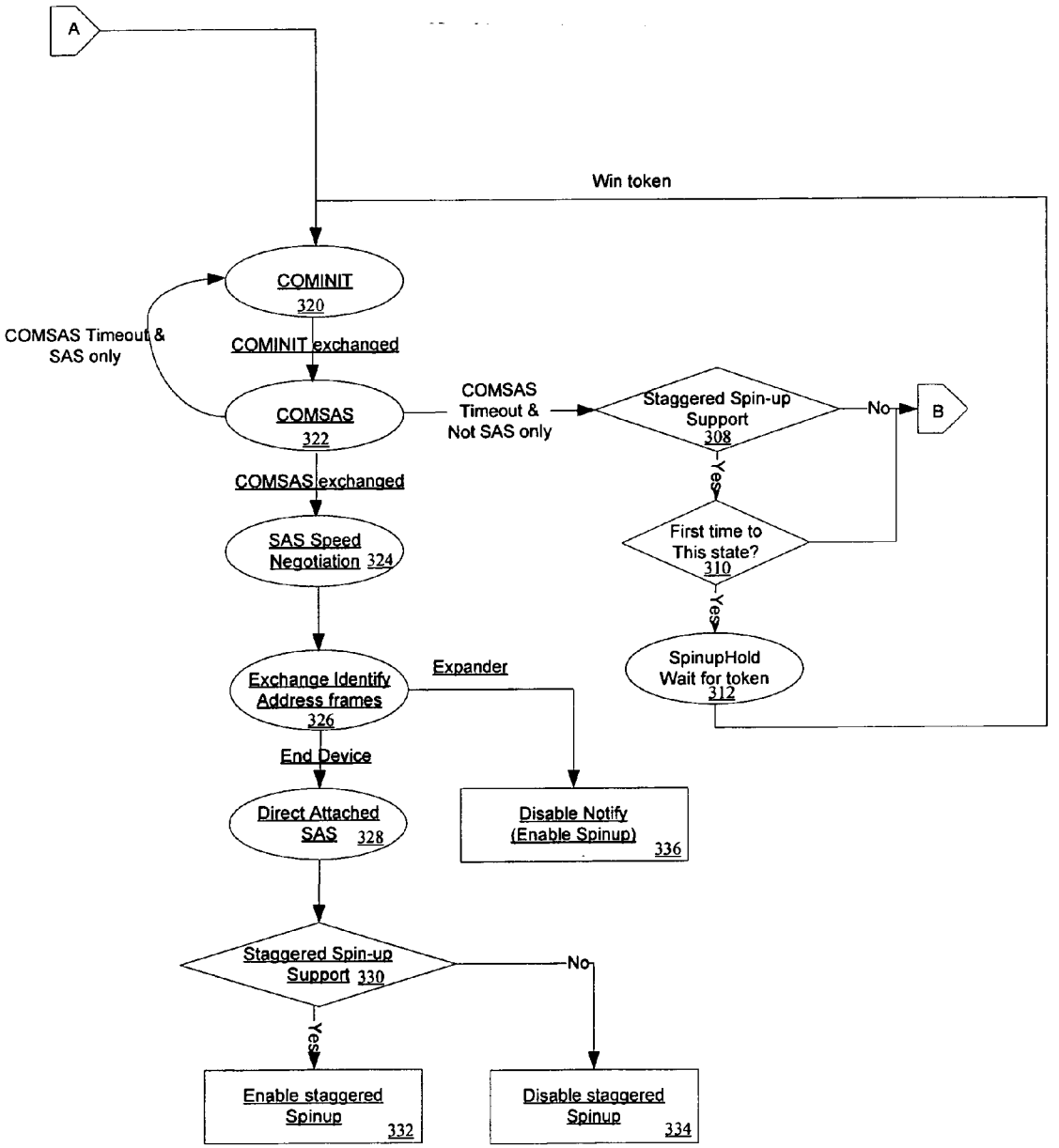

A
Win token
COMINIT
320
COMSAS Timeout & SAS only
COMINIT exchanged
COMSAS
322
COMSAS Timeout & Not SAS only
Staggered Spin-up Support
308
No
B
Yes
COMSAS exchanged
First time to This state?
310
SAS Speed Negotiation 324
Yes
SpinupHold Wait for token
312
Exchange Identify Address frames
326
Expander
End Device
Direct Attached SAS 328
Disable Notify (Enable Spinup)
336
Staggered Spin-up Support 330
No
Yes
Enable staggered Spinup
332
Disable staggered Spinup
334

Figure 3B

HARD DISK DRIVE STAGGERED SPIN-UP MECHANISM

FIELD OF THE INVENTION

[0001] The present invention relates to computer systems; more particularly, the present invention relates to computer system interaction with hard disk drives.

BACKGROUND

[0002] Most of the power used by modern hard disk drives is consumed by the spindle motor. When the hard disk is initially started up, the motor may draw a peak level of power that is more than two times what it takes to keep the disk spinning. While in most cases even the peak start-up power usage is not substantial, there may be an issue when using multiple hard disks that attempt to spin-up simultaneously. Such an occurrence requires a sufficient power supply to withstand this initial demand.

[0003] As a solution to the above-described problem, staggered spin-up is implemented in systems where the host system may spin up the disk drives sequentially. Staggered spin-up significantly lowers design requirements and the cost of the power supply, and avoids overloading of the power supply, reducing the risk of damage to the power supply and the disk drives.

[0004] However, in a traditional host bus adapter (HBA), most of the physical layer (phy) reset sequence state machines are implemented in firmware, leaving staggered spinup a firmware task. The disadvantage of enabling firmware handle phy reset sequence and spin-up is that it adds real time handling requirement to the host CPU, thus slowing down the performance. Further, since host processors are moving further away from the control unit, putting more and more pressure on offloading part or all of the reset sequence state machines in hardware, making firmware implementation of staggered spin-up will become undesirable.

BRIEF DESCRIPTION OF THE DRAWINGS

[0005] The invention is illustrated by way of example and not limitation in the figures of the accompanying drawings, in which like references indicate similar elements, and in which:

[0006] **FIG. 1** is a block diagram of one embodiment of a computer system;

[0007] **FIG. 2** illustrates one embodiment of a Host Bus Adapter coupled to hard disk drives;

[0008] **FIGS. 3A and 3B** is a flow diagram illustrating one embodiment of the operation of staggered spin-up;

DETAILED DESCRIPTION

[0009] A mechanism for the staggered spin-up of hard disk drives is described. In the following detailed description of the present invention numerous specific details are set forth in order to provide a thorough understanding of the present invention. However, it will be apparent to one skilled in the art that the present invention may be practiced without these specific details. In other instances, well-known structures and devices are shown in block diagram form, rather than in detail, in order to avoid obscuring the present invention.

[0010] 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.

[0011] **FIG. 1** is a block diagram of one embodiment of a computer system **100**. Computer system **100** includes a central processing unit (CPU) **102** coupled to an interface **105**. In one embodiment, CPU **102** is a processor in the Pentium® family of processors Pentium® IV processors available from Intel Corporation of Santa Clara, Calif. Alternatively, other CPUs may be used. For instance, CPU **102** may be implemented using multiple processing cores. In other embodiments, computer system **100** may include multiple CPUs **102**

[0012] In a further embodiment, a chipset **107** is also coupled to interface **105**. Chipset **107** includes a memory control hub (MCH) **110**. MCH **110** may include a memory controller **112** that is coupled to a main system memory **115**. Main system memory **115** stores data and sequences of instructions that are executed by CPU **102** or any other device included in system **100**. In one embodiment, main system memory **115** includes dynamic random access memory (DRAM); however, main system memory **115** may be implemented using other memory types. Additional devices may also be coupled to interface **105**, such as multiple CPUs and/or multiple system memories.

[0013] MCH **110** is coupled to an input/output control hub (ICH) **140** via a hub interface. ICH **140** provides an interface to input/output (I/O) devices within computer system **100**. ICH **140** may support standard I/O operations on I/O busses such as peripheral component interconnect (PCI), accelerated graphics port (AGP), universal serial bus (USB), low pin count (LPC) bus, or any other kind of I/O bus (not shown).

[0014] According to one embodiment, ICH **140** includes a host bus adapter (HBA) **144**. HBA **144** serves as a controller implemented to control access to one or more hard disk drives **150**. In one embodiment, hard disk drive **150** is a serial SCSI (SAS) drive. However in other embodiments, hard disk drive **150** may be a serial ATA (SATA) drive. Nevertheless, HBA **144** is capable of controlling either a SAS or SATA device, as well as other device types.

[0015] For spin-up in a serial SCSI (SSP) drive, the host system issues a start-stop unit command (spinup enable) to enable the device for spin up. However, the device is not allowed to start spinning up until a primitive NOTIFY (enable spin-up) is received. In serial ATA (SATA) devices, a device automatically spins up when an out-of-band (OOB) sequence is complete.

[0016] A problem with spin-up of such devices is that each attached device spin-up is not controllable by computer system **100**. For example, if the HBA has eight ports, and if all eight ports are active, with all attached devices located in the same enclosure, spinning up simultaneously requires a power supply that can handle eight times the peak current y at spin-up.

[0017] According to one embodiment, a staggered spin-up mechanism is incorporated in the hardware of HBA **144** to

enable disk drives coupled to HBA **144** to be started up sequentially. **FIG. 2** illustrates one embodiment of HBA **144** coupled to hard disk drive **150**. According to one embodiment, HBA **144** is coupled to eight storage devices **250** within hard disk drive **150** via eight ports.

[**0018**] HBA **144** includes a protocol engine **230**, which represents a link layer to communicate with a SAS/SATA device. Protocol engine **230** includes link layer engines **0-7** corresponding to each of the eight ports, programmable token spacer **245** and token passing logic **240**. The link layer engines controls communication for each operational SAS link. Such communication includes an identification sequence, connection management, and frame transmission requested by the port layer. In one embodiment, the link layer engines each include their own OOB speed negotiation logic.

[**0019**] Further, all eight of the engines communicate with token passing logic **240**. Token passing logic **240** is a shift register with a default one hot encoded value on power up. According to one embodiment, the shift register includes registers SR**0**-SR**7** corresponding to each link layer engine. Programmable token spacer **245** is a counter that may be custom programmed to a value that equals a time difference between the spin-ups of two adjacent devices.

[**0020**] In one embodiment, the minimum value should be set to the minimum spin-up time for the devices. Token spacer **245** operates as a shift enable signal to the shift register. The control signals that are passed from the link layer engines **0-7** to token passing logic **240** are: enable0-7.

[**0021**] According to one embodiment, a link layer engine transmits an enable signal to spin-up its respective device **250**. The particular link layer initiates the transmission of the spin-up whenever the associated register in token passing logic **240** is a logic 1. For example, link layer engine **0** transmits enable0 whenever SR**0** is a logic 1. Subsequently, a logic 0 is shifted to SR**0** causing the logic 1 to be shifted to SR**1**, resulting in enable0 being deactivated and enable1 being transmitted to its corresponding device **250** for spin-up

[**0022**] In a further embodiment, when a link is attached to an expander (not shown), no spin-up is necessary for that particular link because the expander will handle the staggered spin-up itself and will not forward any incoming Notify(enable spin-up) primitives. Therefore, when the link layer detects that the port is attached to an expander, or when it detects that no device is attached, the link layer will transmit a control signal to token control logic **240** to bypass the corresponding shift register component. In this case, NOTIFY primitive may not be sent by the link layer. In one embodiment, firmware may force to mask out a particular link by bypassing the corresponding shift register component. Token passing logic **240** will send one token to the link layer at a time, guaranteeing one spin-up at a time.

[**0023**] **FIGS. 3A and 3B** is a flow diagram illustrating one embodiment of a reset sequence at a link layer engine supporting staggered spin-up. Referring to **FIG. 3A**, the process begins in the reset state **302**. At decision block **304**, the link layer engine determines if it supports only an SATA mode. If so, the link layer engine enters a COMRESET state **306** where it waits for a COMINIT/COMRESET exchange.

[**0024**] At decision block **308**, it is determined if staggered spin-up is supported. If staggered spin-up is supported, it is determined, at decision block **310**, if this is the first time this state is entered. If so, the link layer engine enters Spinup-Hold state **312** to wait for the token. When a token is acquired, the link layer engine goes back to the COMRESET state **306**.

[**0025**] If not the first time to enter this state, or staggered spin-up is not supported, the link layer engine goes to COMWAKE state **314**. At processing block **316**, the link layer engine allows reset of the OOB/Speed Negotiation to finish. The associated device then spins up automatically.

[**0026**] If at decision block **304**, it is determined that not only SATA is supported, the link layer engine enters COMINIT state **320**. Referring to **FIG. 3B**, a COMSAS state **322** is entered after COMINIT is exchanged. If COMSAS state **322** detects a timeout and SATA support is assumed, control is returned to decision block **308** where it is determined if staggered spin-up is supported (**FIG. 3A**)

[**0027**] If COMSAS state **322** detects that a timeout occurs and only SAS is supported, the engine link layer goes back to the COMINIT state **320**. Otherwise COMSAS is exchanged and a SAS speed negotiation state **324** is entered. Subsequently, the engine link layer enters a state **326** where identify address frames are exchanged. If an expander is present, notify is disabled and spin-up is enabled, processing block **336**.

[**0028**] Otherwise from state **326**, the engine link layer enters a direct attached SAS state **328**. At decision block **330**, it is determined whether staggered spin-up is supported at the engine link layer. If supported, staggered spin-up is enabled, processing block **332**. If staggered spin-up is enabled, the enable signal that goes to token control logic **240** gets set. NOTIFY primitive will be sent on this link. If not supported, staggered spin-up is disabled, processing block **334**. If staggered spin-up is disabled, the control signal is cleared. The token control logic will bypass this node, resulting in no NOTIFY primitive being sent.

[**0029**] The above-described staggered spin-up mechanism greatly reduces power supply requirements. In addition, the mechanism provides a stand alone serial interface solution to support staggered spin-up power management and eliminates firmware control of staggered spin-up, which adds real time handling requirement for the host processor. Further, the mechanism eliminates the requirement of a local microprocessor at the HBA, which reduces the design cost to support staggered spin-up.

[**0030**] Whereas many alterations and modifications of the present invention will no doubt become apparent to a person of ordinary skill in the art after having read the foregoing description, it is to be understood that any particular embodiment shown and described by way of illustration is in no way intended to be considered limiting. Therefore, references to details of various embodiments are not intended to limit the scope of the claims, which in themselves recite only those features regarded as essential to the invention.

What is claimed is:

1. A host bus adapter (HBA) comprising:

one or more link layer engines;

one or more ports, each of the one or more ports associated with one of the one or more link layer engines; and

token passing logic having a shift register associated with each of the one or more link layer engines, wherein a first link layer engine enables a first storage device coupled to an associated port to spin-up whenever the first link layer engine detects that a first shift register has a first value.

2. The HBA of claim 1 wherein a second link layer engine detects that a second shift register has a second value whenever the first link layer engine detects that the first shift register has the first value.

3. The HBA of claim 2 wherein the second link layer engine disables a second storage device coupled to an associated port whenever detecting that the second shift register has the second value.

4. The HBA of claim 2 wherein the second link layer engine enables the second storage device whenever detecting that the associated shift register has the first value.

5. The HBA of claim 2 further comprising a token spacer to provide a value to the token passing logic.

6. The HBA of claim 5 wherein the value represents a time difference between the spin-up for the first device and the spin-up for the second device.

7. The HBA of claim 5 wherein the value is programmable.

8. The HBA of claim 2 wherein a third link layer engine transmits a control signal to the token passing logic to bypass a third shift register if a third port is coupled to an expander.

9. A method comprising:

a first link layer engine detecting a first value at a first shift register associated with the first link layer engine;

a second link layer engine detecting a second value at a second shift register associated with the second link layer engine; and

transmitting a first enable signal to a first storage device associated with the first link layer engine to initiate spin-up of the first device.

10. The method of claim 9 further comprising:

the first link layer engine detecting the second value at the first shift register;

the second link layer engine detecting the first value at the second shift register; and

transmitting a second enable signal to a second storage device associated with the second link layer engine to initiate spin-up of the second device.

11. The method of claim 10 further comprising transmitting a value indicating a time difference between the spin-up for the first device and the spin-up for the second device.

12. The method of claim 10 wherein the value is programmable.

13. The method of claim 10 further comprising:

a third link layer engine detecting an expander coupled to a port associated with the third link layer engine; and

third link layer engine transmitting a control signal to bypass a third shift register.

14. A system comprising:

one or more storage devices; and

a host bus adapter (HBA), coupled to the one or more storage devices, having:

one or more ports, each of the one or more ports coupled to an associated storage device;

one or more link layer engines, each of the one or more link layer engines associated with one of the one or more ports; and

token passing logic having a shift register associated with each of the one or more link layer engines, wherein a first link layer engine enables a first storage device coupled to an associated port to spin-up whenever the first link layer engine detects that a first shift register has a first value.

15. The system of claim 14 wherein the HBA includes a second link layer engine to detect that a second shift register has a second value whenever the first link layer engine detects that the first shift register has the first value.

16. The system of claim 15 wherein the second link layer engine enables the second storage device whenever detecting that the associated shift register has the first value.

17. The system of claim 14 wherein the HBA further comprising a token spacer to provide a value to the token passing logic.

18. The system of claim 15 wherein the HBA further comprises a third link layer engine to transmit a control signal to the token passing logic to bypass a third shift register if a third port is coupled to an expander.

19. The system of claim 17 wherein the value is programmable.

20. The system of claim 17 wherein the value represents a time difference between the spin-up for the first device and the spin-up for the second device.

* * * * *