To: T10 Technical Committee
From: Rob Elliott, HP (elliott@hp.com)
Date: 10 September 2002
Subject: T10/02-360r0 SAS spin-up

Revision History

Revision 0 (10 September 2002) first revision

Related Documents

sas-r01b - Serial Attached SCSI revision 01b
spi5r02 - SCSI Parallel Interface 5 revision 2

Overview

Parallel SCSI disk drives offer a variety of mechanisms to control spin-up, all controlled by the RMT_START and DLYD_START pins on the SCA-2 connector. Defined behaviors are:
a) spin-up automatically after power on;
b) spin-up automatically after power on after delaying for (up to 12 seconds) * (the SCSI ID assigned by the SEL_ID pins). This means that a drive with SCSI ID 0 powers on immediately, while a drive with SCSI ID 7 waits up to 84 seconds;
c) spin-up under software control with START STOP UNIT.

The Serial Attached SCSI connector lacks the pins to replicate these features, and SAS does not have enclosure-assigned addresses on which to base any form of delayed spin-up. Therefore, only START STOP UNIT is currently supported.

Serial ATA disk drives, in contrast, will spin-up automatically after power on (specifically, after the phy reset sequence completes).

Enclosures are often designed with power supplies that cannot tolerate all their disk drives spinning up at the same time.

Problem

Serial Attached SCSI offers the ability to connect more initiators and more drives than parallel SCSI. The initiators are not guaranteed to have software coordinating access to the drives; e.g. they could be separate servers each using a dedicated disk drive for booting. With software control of spin-up, they could all choose to spin-up drives at the same time (e.g. if they all boot simultaneously).

To avoid more drives spinning up than an enclosure supports, the enclosure needs some say in spin-up sequencing.

Even after initial power on, independent initiators are not prevented from issuing causing simultaneous spin-up - they could all issue START STOP UNIT requesting a START at the same time. It would be very difficult for the enclosure to intercept these SCSI commands and delay them somehow.

Suggested Changes

Create a new SPINUP primitive. Immediately after power on, it is used to trigger automatic spin-up; afterwards, it interacts with the START STOP UNIT command to delay when software-requested spin-up actually occurs.

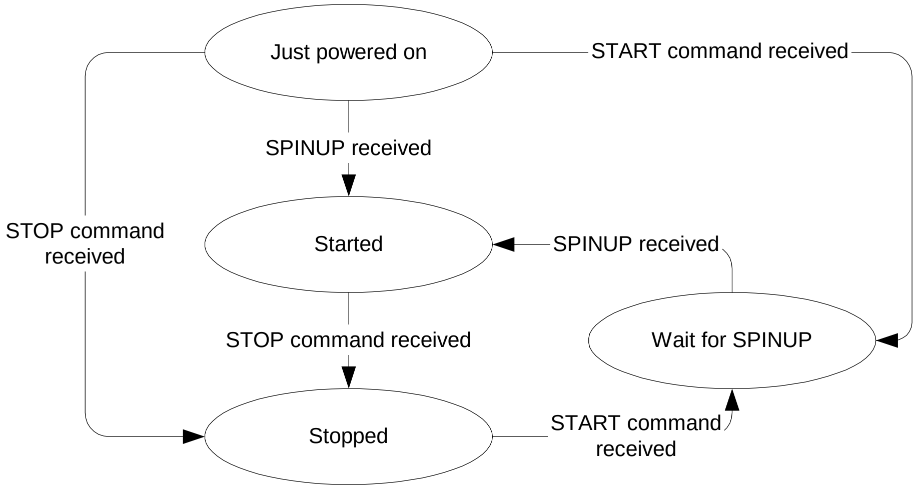
The definition of SPINUP would be:

a) after power on, if the drive has not received a STOP from software, spin-up automatically;

b) after power on, if the drive has received a STOP from software with a subsequent START, spin-up. Software's START request is effectively delayed until the SPINUP arrives;

c) after power on, if the drive has received a STOP from software without a subsequent START, do nothing (only SPINUPs before the first STOP automatically cause spin-up).

This state diagram shows how the target interprets the SPINUP primitive and the START STOP UNIT command requesting either START or STOP:



The enclosure shall generate SPINUP after power on whenever it wants to allow the disk drive to spin-up. It may choose to rotate SPINUP across all its ports (distributing it to N ports at a time if the enclosure power supply is capable of powering up N drives at a time).

SPINUP would be treated like an ALIGN primitive with special meaning. It shall not be forwarded through expanders; it only exists on a physical link. SPINUP shall be sent instead of whichever ALIGN, ALIGN(1), ALIGN(2), or ALIGN(3) was normally going to be sent. This requires the target decode the SPINUP primitive in logic before its elasticity buffers (similar to how ALIGN and ALIGN(2) are differentiated for OOB signals).

Expander devices and initiator devices shall generate SPINUP whenever attached to target devices (devices that report any target protocol support in the incoming IDENTIFY address frame). The selection of when/how often to send SPINUP is vendor-unique. An expander device may allow this timing to be configured by a NVROM, or may involve more complex interaction with the power supply. If a device has no vendor-unique controls, it should default to sequencing SPINUP from one port to another every 12 s. (constantly rotating through all its active phys) [or, we could recommend a default of SPINUPs very often to all phys]

Resets after power on (SCSI level LOGICAL UNIT RESET, running a new phy reset sequence, or running a hard reset) shall not cause spin-up automatically. The one-time automatic spin-up is only based on true power on. [assuming that drives do not spin down on resets]

Multiported targets shall treat SPINUPs from all ports the same. SPINUP on port A can serve as a wakeup for a START command received on port B, for example.

Discussion

Pros: gives SAS a better spin-up control capability than parallel SCSI
Cons: requires SCSI drive firmware implementation of the START STOP UNIT command to not spin-up until a SPINUP is received, requires pre-elasticity buffer primitive decode

Other alternatives discussed:
1. No change for SAS revision 1 rules. Leaves open a multi-initiator problem that we’re more likely to encounter with SAS than with parallel SCSI. SAS drives will suffer slower boots, since software has to issue the START STOP UNIT itself. SATA drives may be spun-up faster (depends on expander design and software design, of course).

2. Drives spin-up automatically after OOB like SATA. If OOB is stopped before completion as done with SATA, the initiator or expander doesn’t know if a target, initiator or expander is attached and if delay is necessary. “Stopping” after OOB requires new rules like withholding the outgoing IDENTIFY or blocking connection requests. Any such solution doesn’t prevent software from spinning-up all drives later.
3. Only honor SPINUP one time after power on. Software can still foil plans and send START STOP UNITs to all drives at once later on.
4. Only honor SPINUP before the first connection. Limits the amount of discovery allowed to SAS expander setup. Want to allow INQUIRY, mode page access, and firmware upgrades before spin-up.
5. Only honor SPINUP outside connections (so it’s an inserted primitive rather than an ALIGN replacement). Some environments may have longstanding connections, and SPINUP would never be seen.