"More Discussion of Tapes in PLDA" T10/97-155r1

9703024

1. Introduction

Here is another attempt at putting together a "what to do about tapes on Fibre Channel" proposal. This version is modified based on discussion that took place during the SSC meeting during the March 1997 T10 week.

I had wanted to keep the introductory material brief, but it became clear at the meeting that most of the confusion is in that material, so it's longer in this case.

Note that there are really two configurations where this topic arises:

i. Native Fibre Channel tape drives.
ii. Native Fibre Channel subsystem controllers (e.g. RAID controllers) that have the capability of having a tape drive behind them. This drive could be a regular SCSI tape drive. Fibre Channel needs to have a tape-oriented protocol on the FC connection to the host even if no native Fibre Channel tape drive is ever built.

2. Overview of the Fibre Channel Tape Problem

Tape devices have different performance requirements than disks. The special characteristics of tapes in an FC-AL environment are summarized as follows:

a. If a tape command or data transfer fails on the interconnect, the recovery requires more than simply the reissuance of the command. The operating system driver software must manage the position of the media by issuing a sequence of repositioning commands in addition to reissuing the failed I/O command. This code is in SCSI tape drivers now, but the mechanical process required to complete the recovery may be time consuming.

Note the distinction between "the application" (user's FORTRAN program) and "the driver" (operating system device driver). The user's program is not supposed to worry about repositioning after an interconnect data error.

Also note that the driver has two parts: "the class driver" (knows about tapes, not interconnects) and "the port driver" (knows about interconnects, not tapes). A goal is to keep these 100% distinct.

b. Using the SCSI command timeout to detect errors is generally unacceptable because the timeout value must be set to a large number (e.g. 10 minutes) to enable normal tape device operation. The timeout method may be acceptable if the error rate at the physical level is low enough so that the timeout is only excercised once or twice a day.

c. When devices are swapped on an FC-AL loop the loop signal is disrupted. It may not be possible to predict when this will occur, but in some environments many devices may be swapped in a day.

d. The FC-AL loop may under normal conditions experience fairly frequent random bit errors. A normal parallel SCSI bus experiences errors at an extremely low rate--weeks may pass between parity errors. It is not known how frequently bit errors will occur on a normally operationg FC-AL loop. Worst-case calculations indicate that hardware complying with the standards may deliver an error bit every 10 seconds.

One may argue what the delivered error rate will be. However, in order to minimize risk at the system level, the PLDA profile must protect against the worst case. The following is based on that assumption.

A secondary goal is to avoid the introduction of Class 2 as a special case for tapes. This is particularly important in the case of

subsystem controllers that must support both disk and tape device models. How is the driver to know whether to send a given INQUIRY command using Class 2 or Class 3? Must the driver handle INQUIRY commands differently from READ or WRITE commands?

The best place to fix the tape problem is at the FCP level as described in PLDA. FC-PH and SCSI are long-established, and changes to SCSI driver software or FC-PH hardware are not desireable. Furthermore, it has already been agreed by the owner of FCP that FCP could be changed if a need can be demonstrated. Small changes to FCP and PLDA cause the minimum amount of disturbance to the status quo.

3. Reliable Tape Transfers to Be Constrained in Size

My previous contention was: It is widely agreed (not universally) that ALL tape transfers may be classified as one of:

a. Transfers where data integrity is required, and where a maximum of 64kBytes will be transferred in any SCSI I/O command, or

b. Transfers where bulk data is being moved and a data error should be ignored, and where the maximum transfer size may be greater than 64kB.

This contention was rejected by the committee. Therefore any solution must handle the case of very long transfers done by a single SCSI command.

4. Overview of Proposed Solution

During the meeting the original proposal was modified so as to add, for READs, what amounts to an FCP-level acknowledgement for every sequence. This can be though of as an "FCP ACK 1".
(In FC terminology, ACK 1 is "acknowledge receipt of one frame".
ACK 0 is "acknowledge receipt of all frames of a sequence".
ACK n is "acknowledge receipt of n frames".)

A new FCP information unit FCP_CONF is needed to send this acknowledgement or confirmation. This allows the initiator to request retransmission of data if a transfer fails, and does not involve the user's application program in the retransmission.

Under this proposal, transfers would look like this:

========
WRITE: Transfer of "n" DATA sequences. Each DATA below is one sequence.

```
Initiator          Target

FCP_CMD ---------->
        <---------- FCP_XFR_RDY
            The target tells the host how much data it can
            accept before another FCP_XFR_RDY will be needed
            Say it's two sequences in this example
DATA  a ---------->     This DATA sequence transferred successfully
DATA  b ----------> This DATA sequence transferred successfully
        <---------- FCP_XFR_RDY
DATA  c ---------->
DATA  d -----X.....     Error occurs at "X"
                        Error is detected by target using sequence count
                        All further frames are ignored
                        Target waits RA_TOV to age any pending frames
        <---------- FCP_XFR_RDY
                        With offset set back to "c"
DATA  c ---------->
DATA  d ---------->
        <---------- FCP_XFR_RDY
```