Date: December 5, 1988
From: Jim McGrath, Quantum
To: X3T9.2
Subject: Terminate Immediate

X3T9.2/88-162

There is currently a proposal that would allow an initiator to terminate immediately an active command and receive residue information. While the standard currently allows such an operation by the sending of an ABORT message, followed by a REQUEST SENSE command, the application driving this requirement requires a higher performance solution. Specifically, the initiator is a caching host bus adaptor that is prefetching data from a SCSI disk via READ commands and copying back write cached data using WRITE commands. To minimize performance losses due to loss of disk revolutions these commands should transfer large quantities of data (as opposed to several shorter commands). However, if the host requires access to data not in the host bus adapter cache, then it must access the SCSI disk quickly. These two requirements conflict unless the long transfers can be quickly aborted and residue information (especially in the case of disk writes) be returned.

However, the TERMINATE IMMEDIATE message solution offered is not necessary nor desirable. First, the earlier assumption that breaking up long disk transactions into multiple shorter transactions would reduce performance by requiring the disk to take a revolution (16 ms) between commands is not valid for modern SCSI disks. Most new disks are implementing some form of prefetching. Thus breaking up a single large read request into several smaller read requests can be done without a loss in performance if the target is sufficiently intelligent.

To give a lower bounds on the size of the read requests consider a SCSI disk with a time from selection to the start of read of 500 us, and from the end of the disk read to sending status and COMMAND COMPLETE of 300 us. Assuming that the host bus adapter can then issue the next command in 200 us, then the amount of time the disk is not reading between commands is 1 ms. Thus prefetching at least 1/16 of a track (or, alternately, imposing a logical interleave of 1/16 of a track) allows a SCSI disk to transfer data to the host bus adapter with small commands just as quickly as with larger commands. For concreteness,

assume a track has 32 sectors. Then single sector (512 byte) reads followed by a prefetch of 2 sectors (quite a common scenario) are sufficient to substain the throughput at the disk's maximum rate.

Disk writes present a more difficult case, since you have no analog to prefetching. Here the disk can "hyperwrite" (issue COMMAND COMPLETE upon receipt of the data in the buffer and before writing it to disk) to overlap disk writes and the receipt of data for the next write. Thus to be safe the transfer size must be atleast equal to 1/16 of a track. This is still only 1Kbytes for 32 sector tracks; 2Kbytes for 64 sectortracks. Both are well within the range of UNIX blok sizes. Note once again that a logical interleave could be an alternative solution.

Both the read and (especially) the write scenerios are simplified tremendously if command queuing is implemented. Here the SCSI disk can use the knowledge of the next few commands queued at the drive to eliminate both the timing requirements and the write caching

requirement. With command queuing it is only required that the bus activity (sending the command, sending/receiving data, receiving status) take less time than the disk activity (reading/writing the disk, head and cylinder switching). Given the use of 4MB/sec synchronous transfer rates, this requirement is easily met if the transfer sizes are 1.5Kbytes (32 sectors/track) or 2.5Kbytes (64 sectors per track). Once again, since UNIX systems are usually the application target, frequently with system block sizes of 4Kbytes, this poses no problem

The second difficulty is that the proposal calls for returning residue information in the messaging system. This poses many architectural problems that I am sure others will comment on.