Sun Microsystems, Inc.
2550 Garcia Avenue
Mountain View, CA 94045
415 960-1300

July 29, 1997

John Lohmeyer
Chairman, T10
4420 ArrowsWest Drive
Colorado Springs, CO 80907-3444

% Sun Logo for

Subject: REQ/ACK Mismatch

Dear Mr. Lohmeyer:

Bill Ham asked me to write up a paragraph for the EPI group discussing the problem of REQ/ACK mismatch and double-clocking on two critical signals on the SCSI bus, request and acknowledge.

I'd like to thank Gerry Houlder and Dan Smith from Seagate and Ting-Li Chen from Qlogic for their help

There are two cases for each mismatch of data flowing in opposite directions. An offset counter, which does not count down on either the target or the initiator side, will cause a time-out and a reset on the SCSI bus. We know from the SCSI-2 specification that the REQ/ACK offset on a synchronous data transfer defines the maximum number of REQ pulses that can be sent by the taget in advance of the number of ACK pulses received from the initiator. For successful completion of the data phase, the number of ACK and REQ pulses shall be equal.

1) One direction is the READ and the data is presented on the bus by the target and validated by the REQ pulse issued by the target:

   a) One or more additional REQ(s) will cause a data transfer time-out due to glitches on the REQ line.

      The right number of ACK pulses from the initiator will zero the target offset counter. The initiator offset counter will not count down to zero because it will have registered more REQ pulses than agreed upon and will not have matched them with the negotiated number of ACK pulses.

      The initiator will time out and might post a message " data overflow ".

   b) The second case is when one or more REQ pulses go undetected, i.e. not counted by the initiator offset counter. The initiator counter will count down to 0 and wrap around. It will time out and might post the message "data underrun".

2) On a WRITE, the data flows in the opposite direction from an initiator to a target. The date is presented on the bus by the initiator and validated by its ACK pulse.

   a) One or more ACK pulses on the ACK line, due to the noise or glitches, will cause the target offset counter wrap around and stall at one of the highest offset position. The will wait for more ACK pulses to come and the initiator willeventually time-out.

   b) In the absence of enough ACK pulses or undetected ACK pulses, the target will not quite count down and the offset counter will not reach 0 and it will time out.

There are two types of the REQ/ACK mismatch or miscalculation.

If sequencers or other logic circuits on either the initiator or the target side are fast enough to finish the data transfer cycle before the additional interlocking pulse(s) arrive, then such a transfer becomes valid. This would probably cause a double-clocking and an unnoticed transfer of the same data twice. The only way to catch such an error would be in the case of a parity error. In the absence of a parity error or in a case of undetactable dual parity error, only CRC or ECC scheme would be of any help.

If sequencers or other logic circuits on either side are not fast enough to finish the data transfer cycle before the additional interlocking pulse(s) arrive, i.e. if the instruction time is longer than the transfer period, then such a transfer will cause the above mentioned time-out.

Sincerely,

Vit F. Novak
Sun Microsystems