To: **T10 Membership**
From: **Lawrence J. Lamers, Vincent Bastiani**
Subject: **Groundwork for Expander Communication**
Date: **Wednesday, March 31, 1999**

This proposal lays the groundwork for expander communication in the next version of SPI. It does this by providing a minimum pulse width for asynchronous non-data transfers

For expander communication to function correctly all devices on a multi-segment SCSI bus need to be able to snoop the bus. In order to ensure that this happens the data must be valid for a minimum period of time. It has been suggested that this time be on the order of 50ns.

One method for achieving this is to specify a minimum assertion period for the REQ and ACK signals. This splits the burden between initiators and targets but doesn't address legacy devices.

For MESSAGE IN and STATUS phases the data is valid from the assertion of REQ to the assertion of ACK. Therefore if the initiator qualifies the assertion of ACK for a period of time that ensures the minimum data valid time.

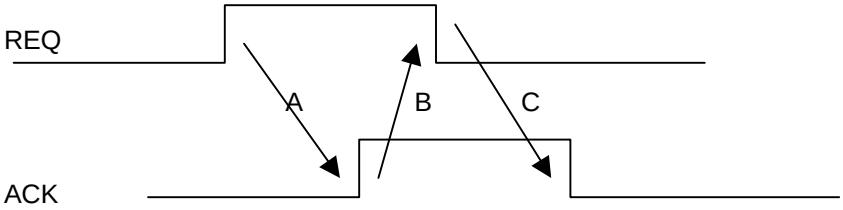
For COMMAND and MESSAGE OUT phases the data is valid from the assertion of ACK to the negation of REQ. However the initiator may keep the data valid as long as ACK is asserted. Therefore if the initiator keeps the data valid until the negation of ACK that ensures the minimum data valid.

The proposal is to specify a minimum REQ assertion to ACK assertion period of 50ns on data transfers to the host; and a minimum ACK assertion period of 50ns with data being valid until ACK is negated on data transfers to the target.

Minimum pulse width Target to Initiator

The REQ pulse width in asynchronous transfer when data is moving from the target to the initiator is determined by the time it takes for the initiator to respond with the assertion of ACK.

The diagram below illustrates the handshake.



The assertion edge of ACK is triggered by the assertion edge of REQ (Time A). The negation edge of REQ is triggered by the assertion edge of ACK (Time B) and the negation edge of ACK is triggered by the initiator seeing the negation of REQ (Time C).

Times B and C could be combinatorial in nature. Time A is not combinatorial since the initiator looks at the data before deciding what to do and hence there is logic involved. If a synchronizer is used the minimum time is two clocks or 50 ns for time A. The duration of A is the most important element in determining the minimum length of REQ in an asychronous cycle. If the receiving device is slow returning ACK then REQ is asserted and the data valid for that duration.

However if the assumption is made that A, B, C are all combinatorial the minimum REQ pulse that could be produced on bus of zero length is:

REQ Min Pulse:
Assert REQ → cable delay → IN Buf of INIT → Core Logic of INIT → Out BUF of INIT → Cable Delay
→ In Buf of TARGET → Core Logic of TARGET → Out BUF of TARGET.

If we assume representative timings for the various delays as follows:

IN BUF = 1.5 ns; OUT BUF = 3.2 ns; Core Logic = 0.7 ns

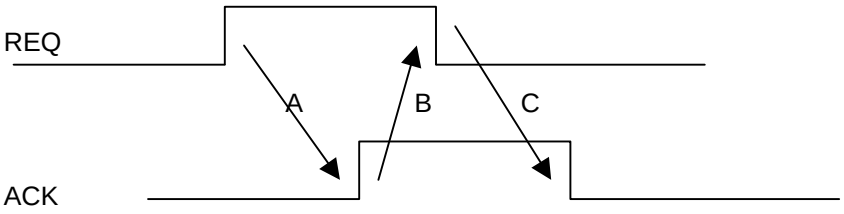
The minimum REQ Pulse could be:

Assert REQ → 0 ns → 1.5 ns → 0.7 ns → 3.2 ns → 0 ns → 1.5 ns → 0.7 ns → 3.2 ns = 10.8 ns.

Minimum pulse width Initiator to Target

The REQ pulse width in asynchronous transfer when data is moving from the initiator to the target is determined by the time it takes for the target to respond with the negation of REQ.

The diagram below illustrates the handshake.



The assertion edge of ACK is triggered by the assertion edge of REQ (Time A). The negation edge of REQ is triggered by the assertion edge of ACK (Time B) and the negation edge of ACK is triggered by the initiator seeing the negation of REQ (Time C).

Times B and C could be combinatorial in nature. Time A is not combinatorial since the initiator asserts the data before asserting ACK. The duration of A is not important in determining the data valid time. If the target is slow deasserting REQ then the data is valid for the duration of ACK and REQ asserted.

However if the assumption is made that B and C are combinatorial and that data is valid for the duration of ACK then the minimum data valid pulse that could be produced on bus of zero length is:

Data Valid:
Assert ACK → cable delay → IN Buf of TARGET → Core Logic of TARGET → Out BUF of TARGET → Cable Delay → In Buf of INITIATOR → Core Logic of INITIATOR → Out BUF of INITIATOR.

If we assume representative timings for the various delays as follows:

IN BUF = 1.5 ns; OUT BUF = 3.2 ns; Core Logic = 0.7 ns

The minimum REQ Pulse could be:

Assert ACK → 0 ns → 1.5 ns → 0.7 ns → 3.2 ns → 0 ns → 1.5 ns → 0.7 ns → 3.2 ns = 10.8 ns.