

Minutes of FCL Flow Control sswg (3/6/97)

The FCL flow control sswg conducted a conference call on Thursday, March 6th, from 12 to 2:00 PM. The topic of discussion was flow control buffer requirements for the different proposals (Aaron and RIP Torn). The following people participated:

Charles Binford Symbios  
 David Ford Clariion (chair)  
 Ian Judd IBM  
 Bill Martin HP  
 Steve Morris Adaptec  
 Curt Ridgeway LSI  
 Jeff Stai Brocade  
 Jeff Williams Western Digital

I had generated a Powerpoint presentation for the conference call. You may obtain a copy at Orca's anonymous ftp server:

<ftp://ftp.orca.com/ftp/pub/fcsswg/buffer1.ppt>

We started out discussing point-to-point flow control as defined in PH. For review, Point-to-Point Flow Control is a Credit based buffer-to-buffer flow control mechanism where only each pair of adjacent Ports along the path between the source and destination Ports inclusive administer the flow control policy.

We examined a 10 km. link and derived what amount of buffering would be required to maintain streaming at full link bandwidth (100 MB/sec) between two Ports at the opposite ends of the 10 km. link. For optical fibre this translates to 100 us. round trip (5 ns/meter). We derived the length (in time) of a 2048 byte frame as 20.84 uses. We did not consider frames with an optional 64 byte header. Charles Binford correctly pointed out in followup email that the link rate is 1.0625e9, not 1e9, which is 6.25% faster than a Gbit/sec. In addition, we did not account for the 6 Idles between frames. The correct time length for a 2K frame with 6 preceding Idles is 19.84 uses.

We then derived the time between transmission of the first frame and receipt of the first R\_RDY as:

$$\text{RT link (100 us) + node delay (.24 us) + 2 * R\_RDY processing time (xmter and rcver - .2 us) = 100.44/19.84 = 5.0625 frames}$$

Thus, the transmitter would have to be allocated a credit of 6 2K buffers to maintain streaming (although 5 would result in a negligible drop in throughput). This analysis assumes that the receiver can send R\_RDY as soon as first frame (of the sequence) is received. This requires either additional buffers available beyond login credit or simultaneous load and unload of receive buffers.

Ian Judd asked if transmit buffer requirements were considered. I replied that I thought transmit buffer requirements would be similar for the various proposals and I did not include transmit buffer costs in any analysis.

As the frame size decreases, the pipe between transmitter and receiver can be filled with more frames, which may require more credit to avoid underrun. This can be mitigated by using variable sized receive buffering, which will have a higher dynamic value of credit for shorter frames. On

addition, we were examining streaming without credit underrun, for which max sized frames will be used to increase throughput.

This analysis holds for simple point-to-point links. For Nearest Neighbor (NN) flow control as defined in RIP, one additional full size buffer is required for the cut through or bypass FIFO (not needed for Aaron). (For review, the definition of NN - a Point-to-Point buffer-to-buffer flow control mechanism with addressable flow control primitives.) Thus, seven 2K buffers are required per node for Nearest Neighbor to maintain streaming over 10 Km. links. This analysis is not valid for Aaron or Torn. We examined Aaron next.

We then examined Aaron flow control, which I labeled as a Store and Forward flow control mechanism to highlight the differences with point-to-point and Nearest Neighbor. With a store and forward architecture, a frame is buffered and cut through the upstream node if it is forwarded further upstream (i.e., it is not addressed to the upstream neighbor). A buffer is freed and an R\_RDY is sent downstream when an ack is received from the next upstream neighbor. This ack is sent when the 2nd upstream node has received the entire frame and checked the CRC. Thus, both the round trip link time of the immediate upstream neighbor and its neighbor determine the amount of buffering or credit required to maintain streaming.

For Aaron, the total time required between transmission of a frame and receipt of R\_RDY is;  $RT_{link1} + RT_{link2} + \max_{frm} \text{ time} + 2 * \text{ node delay} + \text{ ack gen time} + \text{ ack-R\_RDY conversion time} + R\_RDY \text{ processing time}$ . For 10 km links, this time will be over 200 us, which equates to a credit of 11 2K buffers to maintain streaming. Jeff Williams pointed out that this buffering as required for each direction (clockwise and counter-clockwise links), which equates to 22 buffers or 44 Kbytes.

Ian Judd claimed this analysis of Aaron ignored any benefits that the Aaron ack protocol may have on link error recovery. I replied that, yes, we were just examining flow control issues. Ian countered that it was difficult to consider issues such as buffering requirements for flow control in isolation. I countered that we should attempt to do so to facilitate analysis and understanding of various features of the different proposals. We moved on.

We then considered two types of devices for short and long links. If we assume only 2 max frame sized A/B buffers for devices such as disk drives, then we derived the max link length between devices that can still maintain streaming without credit underrun. The required credit is equal to  $RT_{Link1} + RT_{Link2} + \max_{frame} \text{ time}$  (ignores node and ack/R\_RDY processing delays) = 2 max frames (A/B buffers). By dropping a max frame from each side of the equation, we have  $RT_{Link1} + RT_{Link2} = \max_{frame} \text{ time} = 19.84 \text{ uses}$ . If we assume link1 and link2 are the same length, then  $19.84 / 4 = 4.96 \text{ uses} = 1 \text{ km}$ . (approximate). Thus, if the link length is 1 km. or less, dual or A/B max frame sized buffers will facilitate max streaming rate at 100 MB/sec. with no credit underrun. Longer links could be supported by special devices with more buffers, such as interface cards in disk cabinets, switch/hub ports, or HBAs. With point-to-point flow control, only those devices at either end of long links need to support additional buffering. However, this may require detailed knowledge of the configuration to deploy.

We then examined the Torn buffering requirements for flow control. To review, Torn is a source-destination flow control mechanism. The definition: Source-destination flow control - a Credit based buffer-to-buffer flow control mechanism where only the end points along the path between source and destination - the source Port and destination Port - administer the flow control policy. This mechanism applies only to loop topologies. It requires either a Circuit between source and destination (AL) or addressable flow control primitives.

We first considered the required buffering to maintain streaming in Idle Mode, in which the transmitter assumes Login credit's worth of buffers are available. For a source-destination

mechanism, the round trip time for flow control signaling is now equal to Loop Time instead of the round trip time for an individual link as in point-to-point flow control. We defined the time between transmission of first frame of sequence and receipt of first solicited R\_RDY as the Loop Time. The Loop Time is always greater than a link round trip time!

The credit required to maintain Torn streaming at full bandwidth is configuration dependent. The credit required to maintain streaming is equal to the Loop Time + 2 \* R\_RDY processing time. For a 256 node loop with one 10 Km link and the rest 2 meter links, the Loop Time is equal to 163.47 us, which is = 8.24 max length frames. Thus, a login credit of 9 is required to maintain streaming for this configuration.

Configurations with a longer Loop Time will require an increasingly larger credit value. This Loop Time does not take utilization of the loop or congestion into account. The quoted time of 163.47 us should be interpreted as the minimum time or the idle loop propagation delay. Loop utilization and congestion will increase the Loop Time, which may cause transmitter underrun and inhibit streaming.

This analysis only considers a single transmitter talking to a single receiver. Login credit will have to be allocated to multiple transmitters by a receiver. We did not determine how to quantify this effect.

Next we examined Torn buffering requirements for streaming in Active Mode, in which there is no Login Credit available. A transmitter must wait an average of 1/2 Loop Time for first R\_RDY before it may transmit. This assumption is based on one broadcast R\_RDY in flight per receiver. For the next frame in the sequence the transmitter must wait a full Loop Time for subsequent R\_RDYs from the receiver, at which point it may transmit at full link bandwidth. This assumes R\_RDYs supplied to maintain full bandwidth streaming from then on. Thus, in 1 1/2 Loop Times, a transmitter may send two frames, after which it may stream frames at full link bandwidth. The time to transmit N frames is then  $2 * 1.5 * \text{Loop\_Time} + (N-2) * \text{frame\_time}$ . As the transfer length of the sequence increases, N becomes larger and the 1 1/2 Loop Time startup cost may be amortized over longer transfer lengths.

However, it was pointed out that for many workloads the average transfer length is near 4 Kbytes and N is then equal to two. Thus, the majority of 4K transfers will take 1 1/2 Loop Times.

The time to transmit the initial frame may be reduced by having more than one R\_RDY in flight per receiver. For example, with two equidistant R\_RDYs, the initial delay for the first time is 1/4 Loop Time. This may also reduce the delay when transmitting the second frame of the sequence. For example, with 2 R\_RDYs equidistant from each other on the loop, the second frame will be sent 1/2 Loop Time after the first.

Finally, we examined other buffering issues for further analysis. We discussed whether we should allocate additional buffering to minimize slow drain behavior for Aaron. Although additional buffering may reduce the occurrence of slow drain, in some cases it would only delay its onset. Therefore, it was felt that additional buffering was not the correct solution. We then discussed flow control at the ULP level as a solution. There are drawbacks with this approach. It would require a different mechanism for each ULP supported by Fibre Channel. Reads cannot effectively be controlled by XFER\_RDY, since they are issued by the producer (targets) instead of the consumer (initiators).

We then discussed using class 2 as a flow control mechanism. Bill Martin pointed out that class 2 may have issues when used as an end-to-end flow control mechanism in FCL. In today's Fibre Channel, class 2 credit may be overcommitted and flow control handled by BB\_Credit. However, this would no longer be the case in FCL if we used class 2 as the primary flow control

mechanism. It is likely that in larger configurations such as fabrics a receiver may be forced into over committing EE\_Credit, which is exactly the same problem we were trying to solve in the first place. It was felt we may need some other transport mechanism independent of ULPs to control end-to-end pacing if we chose to address slow drain for Aaron.

The call ended just shortly after 2:00 PM EST.