Simulation of the Performance of XDQRAP under a Range of Conditions

John O. Limb Dolors Sala
Jason Collins David Hartman Daniel Howard

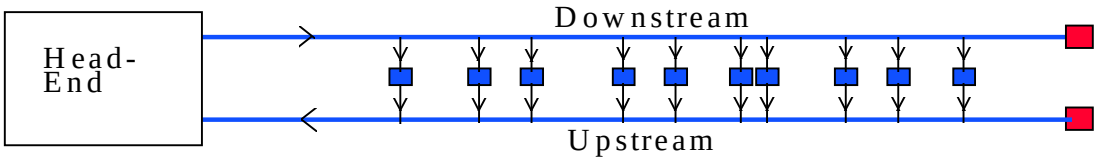
Georgia Institute of Technology *
Atlanta, GA. 30332-0280

1. Overview of XDQRAP Protocol

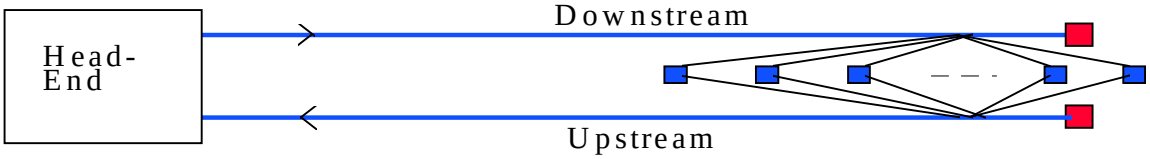
For a more complete description of the XDQRAP protocol see references [1] and [2]. We give here a brief description of a specific version of the protocol, the one that we feel is most appropriate for cable applications. This is the version that we simulated. Many variations are possible. A transmission of a cell proceeds in two steps, first reservation and then transmission. We will first describe the configuration at the physical level and then describe the reservation process and then the transmission process.

1.1 Physical Level

The stations attached to a cable network can be represented from a performance point of view as shown in Figure 1a below. Stations writing on the upstream path cannot sense the channel before writing. They can write randomly or in response to some command from the head-end or some global synchronization signal. We assume that there is an underlying synchronization mechanism that permits stations to write in slots. In practice there will need to be some guard band between slots to accommodate any uncertainty about when a slot starts. XDQRAP inserts delay into the downstream signal of each station to make it appear to be located at the remote end of the cable, as shown in Figure 1b. Spaced between data slots are one or more minislots (including guard bands) that are used to make reservations.



(a) Abstraction



(b) XDQRAP Abstraction

Figure 1 Physical Configuration for XDQRAP

* This work was performed under the sponsorship of Scientific Atlanta through a contract.

1.2 Reservation Process

Packets arriving at a station are segmented into cells and stored in the input queue, AQ. When there are no more packets in the process of seeking a reservation, new packets can be scheduled. These packets enter the reservation queue, RQ and write in one of the minislots located after the current cell. Assume two minislots per cell. A minislot is selected randomly. The minislot travels to the H/E and returns on the downstream line. The station that sent the request knows when to expect the reservation back on the downstream line. If the minislot in which it wrote is not corrupted (i.e. no collision) the reservation was successful and the packet then proceeds to the transmission queue at the station. Note that data transmission can proceed in parallel with the reservation process, since they use separate slots.

Either or both minislots may suffer a collision with two or more stations trying to write in the same minislot. If this occurs the whole system enters into a collision resolution phase in which the time to resolve will depend on the number of stations involved in the collision. If both minislots have collisions the stations writing in the second minislot defer until those stations writing in the first minislot have resolved their collisions. If many stations are seeking to reserve at the same time it would be possible to have both minislots suffer collisions again in the second cycle of the process. Again, those stations colliding in the second minislot drop out until the stations competing for the first minislot resolve. The resolution process is very similar to the tree resolution algorithm proposed as an enhancement to Ethernet [3]. During the contention resolution process all stations must track the process at each stage. Failure to do so, even by a station not making a request, would prevent the station from joining the reservation process at a later stage as it would not know when a cycle was completing and thus when it was a valid point for it to do so. The RQ together with the mechanisms for making and tracking the reservation process is referred to as a Reservation Engine (RE). An RE can only have one request (and transmit) outstanding at a time.

If a line has a round-trip delay longer than the length of a cell, say 4, cells then a station could only use every fourth cell to make a request (because of only one outstanding request) and the maximum bandwidth available to a station would be limited. To overcome this the station replicates the RE so that now a station may have one outstanding request for each RE. In this way a single station can exploit more of the available channel.

Normally messages go through the reservation process and then transmit their data. There is an "Immediate" mode discribed in Section 1.4 where a station may reduce the access delay by transmitting the data at the same time that the request is made (see section 1.4).

1.3 Transmission Process

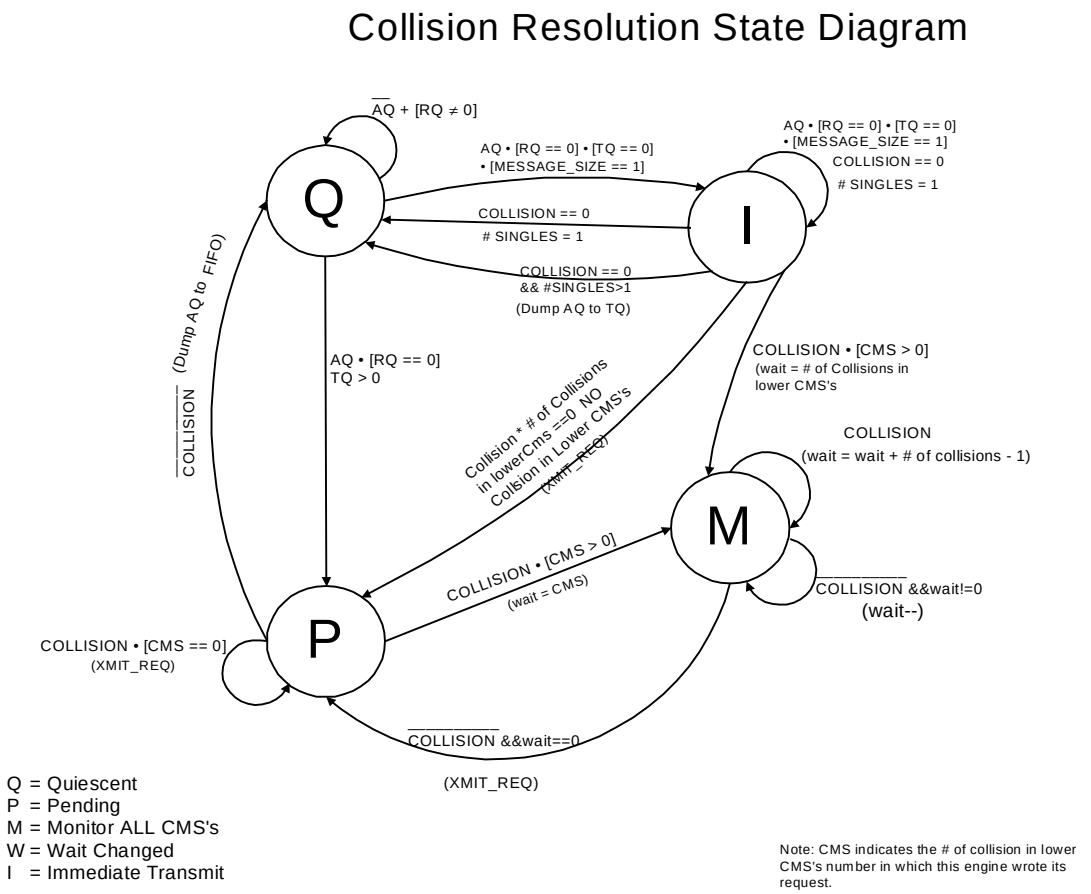
Resolved requests enter the Transmission Queue TQ and await the correct time to transmit. TQ is a global queue in the sense of DQDB. Thus all competing stations must monitor the downstream line to determine when reservations complete so that a count can be made of entries to the TQ that are ahead of and behind the request of the station. It is important that all stations know the exact state of the TQ if they are not to overwrite the data of another station. When all requests ahead of the station have been transmitted the station writes in the next data slot and continues to count down the requests of stations occurring after its own transmission to ascertain the end of the reservation/transmission cycle.

1.4 Data Transmission Rule 1

The protocol permits a station to transmit in a data slot without a reservation in one specific instance - if a packet, of size one cell, enters the system and there are no cells in the global system (i.e. in RQ or TQ). There is no danger of a collision in the data slot of a packet that has received a valid reservation. A collision may result in the data slot if another packet arrives during the same cell time and it is also exercising Data Transmission Rule 1 (DTR1). The station exercising DTR1 also writes in one of the associated minislots. If there is a minislot collision then the stations involved enter contention resolution as before. If both minislots are busy, there could be a collision in the data slot and, again, the stations enter collision resolution. If only one station makes a request under DTR1 then the cell will have been transmitted correctly.

1.5 Comments

This is the outline of the protocol that has been simulated. A detailed description would take far more space. There are some aspects of the simulation that are not completely described in References [1] and [2]. Where there was ambiguity or uncertainty we tried to make assumptions that on one hand would not make the implementation too cumbersome and on the other hand, would lead to an efficient protocol. The station state diagram we used for our implementation is shown in Figure 2.



2. Simulation

The system was simulated in a significant amount of detail, but only at the MAC level. All station and cable queues and delays were implemented within the simulation. It was assumed that the head-end passed the data on without changing it in any way. The simulation is continually evolving. The simulation was done twice, by separate teams, one in C and the other in C++. The way in which the implementations were done was rather different in the way time was handled. One was more akin to a discrete event simulation while the other was more like a discrete time simulation. In virtually all cases the simulations were run for 110,000 events with statistics being gathered after 10,000 events. Except for very high values of delay (when the system is close to overload) the accuracy (standard deviation) of the measurements is usually better than 5%.

We were able to vary a number of parameters:

Number of stations	10, 50, 100
Length of line	2, 3, 6 cells times
Traffic Model	length of burst, duration of burst
Number of minislots	2, 3

We were able to measure a number of variables: the queue distributions, average delays at stations and minislot collisions. There are enough combinations of parameters and variables that we cannot hope to cover the whole space. What we have done is to present a “standard” case and vary each parameter about the standard. Our standard case assumes 50 stations, bursty traffic that consists of 33% bursts that are 11 cells long and 66% that are one cell in length. The line is assumed to be 3 cells long from the furthest station to the head-end and back (i.e. round trip).

3. Results

Figure 3 shows the results of the standard case (graph b). The mean delay is the time from when the cell enters the system (queue AQ) until the time the cell is written onto the upstream channel. A normalized load of 1.0 would mean that every data packet is written in. The delay is quite small up to a load of about 0.8 and the system does not overload until the load is greater than 95%. Notice that performance is not discernibly effected by the number of stations. Although not shown, the system gives about equal access to all stations. Since station-delay, for purposes of reservation, is padded out to an integer value there is no inherent positional advantage in the system.

Figure 4 shows that the length of the system has very little affect. The number of RE’s is, in fact, the round-trip-delay increased to the next integer value. There is a small delay penalty associated with a longer line.

Figure 5 shows the impact of the type of traffic. We are seeing two effects here. One is congestion in the request mechanism. Traffic (1,1) represents short messages that can be transmitted in a single cell. The (2,0) traffic represents fixed length messages of length 2. At 100% load the load on each minislot is 25%, < 33%, therefore, minislot contention is not a limiting factor. Further, since the (2,0) traffic is less bursty than the (11,2) traffic, congestion in the data slots produces greater delay for the (11,2) traffic compared to the (2,0) traffic. The (11,2) traffic has 33% of the messages in bursts 11 cells long and 67% of the messages have a length of 1 cell.

At low loads the (1,1) traffic produces very low delay because DTR1 rule is used successfully for approximately 80% of traffic (at a load of 0.2). The (2,0) traffic, on the other hand, cannot use DTR1 and so the limiting delay is the round-trip-delay, 2x3=6 slots in this case.

Figure 6 explores further the performance of the minislot reservation channel. For the two cases, 3 minislots with fixed-length bursts of size 2 and 2 minislots with fixed-length bursts of size 3 the load on each minislot should be the same and we would expect the percentage of minislot collisions to be very similar. In the lower figure we see that, in fact, the number of minislots is very close at low loads, and diverges a little at higher loads. The total level of collisions is small, and we recommend this for good performance. A figure of less than 4% for minislot collisions is a good goal.

4. References

1. Wu, Chien-Ting and Graham Campbell, "Extended DQRAP (XDQRAP) A Cable TV Protocol Functioning as a Distributed Switch", Proc. 1994 1st International Workshop on Community Networking, pp 191-198, July 13-14, 1994.

2. Xu, Wenxin and Graham Campbell, "A Distributed Queuing Random Access Protocol for a Broadcast Channel" ACM SIGCOMM'93, pp 270-278.

3. Capetanakis, T. “Tree Algorithm For a Packet Broadcasting Channel”, IEEE Transacation Information Theory, Vol, IT-25, pp. 505-515, September, 1979.