An Efficient Scheme for Accommodating Synchronous Traffic in a Cable-Modem Network While Avoiding Segmentation of Asynchronous Packets

Reuven Cohen
HP Israel Science Center[*]

Keywords: Cable-Modem, MAC, MXL, synchronous-service, QoS

Abstract

Cable-modem (CM) technology is being developed to provide high-speed multimedia services to the subscribers' homes over the existing hybrid-fiber-coax (HFC) infrastructure of cable TV networks. The paper proposes an efficient scheme for combining asynchronous and synchronous traffic on the upstream channel of a CM network when segmentation of the asynchronous packets is to be avoided. The new scheme guarantees the synchronous sources the same quality of service provided by the FDDI timed token protocol. That is, a guaranteed average delay between consecutive transmissions, a guaranteed maximum delay between consecutive transmissions, and a guaranteed bandwidth on the upstream channel.

[*]Address: HP Israel Science Center, Technion City, Haifa 32000, Israel. Email: rcohen@hp.technion.ac.il

1 Introduction

Cable-modem (CM) technology is being developed to provide high-speed multimedia services to the subscribers' homes over the existing hybrid-fiber-coax (HFC) infrastructure of cable TV networks. It is widely recognized that cable-modem networks will play an important role in providing economical service access for residential subscribers. The large excess bandwidth available in cable TV networks can provide the communication infrastructure for home interactive services like video on demand, web surfing and video game playing.

In a typical HFC network, cable-modems are connected to a common SCS (signal conversion system, sometimes referred to as head-end) by a tree-and-branch transmission network. A cable-modem is more complex than an ordinary telephony modem, because it has a network interface card which implements some access control protocol to arbitrate the upstream channel among multiple users. The network is divided into a fiber domain which extends from the SCS to a neighborhood, and a coax domain which connects the homes to the fiber. Fiber nodes at the edges of the fiber and coax domains transform the optical signal to an electrical signal. The maximum length of such a network is approximately 50 miles. The homes are located only at the last 20 percents of this distance.

The design of the HFC network forces distinct downstream and upstream channels to be used for communication to and from the home, respectively. In most deployment, the downstream channels operate in the 450-750 MHz frequency band whereas the upstream channels operate in the 5-40 MHz band. Using appropriate modulation techniques, each of the upstream channels is usually capable of carrying around 0.5-3 Mb/s, whereas each of the downstream channels is capable of carrying 3-30 Mb/s. In addition to setting some spectrum aside for upstream and downstream digital transmission, the cable operators need to upgrade existing amplifiers with duplex filters to separate and simplify the upstream and downstream signals.

Each of the upstream channels is shared by the stations (CMs) using a multiple access control (MAC) protocol. Due to the CATV tree-and-branch physical architecture, the stations cannot directly listen to the upstream transmissions of other stations, and cannot detect collisions. However, unlike traditional LAN which are fully distributed, the SCS station of an HFC network can play a major role in coordinating the upstream transmissions stations. The SCS receives all signals of the upstream transmissions and can either repeat them on the downstream channel or process them and inform the CMs about collisions. Several MAC protocols for the upstream channel have been proposed so far (e.g. [2, 4, 7, 8]). The MXL protocol, presented in [7], is unique because it does not require that the IP variable length packets will be broken down into a stream of fixed sized units (cells) at the sending CM and be re-assembled at the SCS after delivery. This property of the MXL simplifies the implementation and reduce the cost significantly. However, it gives rise to some difficulties when synchronous traffic (like voice and video), that arrive at regular intervals and require timely delivery, should be accommodated on the upstream channel along with asynchronous traffic (data packets).

This paper proposes a new scheme for supporting synchronous traffic on the upstream channel of a CM network while avoiding the need to fragment asynchronous packets. The proposed scheme is general enough to be implemented in various CM or non-CM (e.g. cellular or satellite) network architectures. Nevertheless, it will be presented in the context of MXL for which it has been originally designed[1]. In Section 2 the MXL network and MAC proto-

[1]Note, however, that this scheme will not necessarily be adopted for the MXL.

col are briefly introduced. Section 3 describes the problem of accommodating synchronous data on the upstream channel of the MXL. Sections 4 and 5 present the new scheme which guarantees the synchronous stations the same quality of service provided by the FDDI timed token protocol [6]. That is, a guaranteed average delay between consecutive transmissions, a guaranteed maximum delay between consecutive transmissions, and a guaranteed bandwidth on the upstream channel. Section 6 proves the main properties of the new scheme and Section 7 concludes the paper.

2 The MXL Topology and Upstream Access Control

This section presents the main concept of the MXL – an upstream channel MAC protocol designed by HP for forwarding non-fragmented Ethernet packets from the home PC's via the CM's and the SCS to the Internet. More details on the MXL can be found in [7].

The MXL (multimedia transmission link), associates a downstream 6 MHz channel with an upstream 2 MHz channel. Using a 6 bits/symbol 64QAM modulation scheme, which carries one bit for control and 5 bits for payload from each symbol, the 6 MHz downstream channel is divided into a 25 Mb/s downstream data channel and a 5 Mb/s downstream control channel pair. The 2 MHz upstream channel is shared by all the cable-modems (stations) for delivering QPSK-modulated data and control packets to the SCS in a rate of 3Mb/s. Figure 1 depicts the basic architecture of an MXL CM network.

In MXL, a slot is a unit of time on the upstream channel. Its length is represented by a fixed number of downstream control channel mini-slots. This number is programmable by the MXL SCS. The length of the slot includes the time to transmit a control packet upstream, and the round-trip propagation delay between the most distanced stations. The latter distance must be smaller than the length of a control upstream packet. The stations are synchronized such that each transmission arrives at the SCS within the timing bounds of a time slot. The timing of the upstream slots is derived by the stations from counter timing data transmitted by the SCS in each mini-slot on the downstream control channel. In addition to providing these synchronization markers, the SCS transmits on the downstream control channel a short status packet (which uses a single mini-slot) every upstream slot time. The purpose of this short packet is to inform the stations of the status of the next upstream slot. An upstream slot can be either reserved or available. A reserved slot can be used by a single station, to which it has been previously assigned by the SCS. An available (contention) slot is open for contention according to the rules dictated by the contention algorithm.

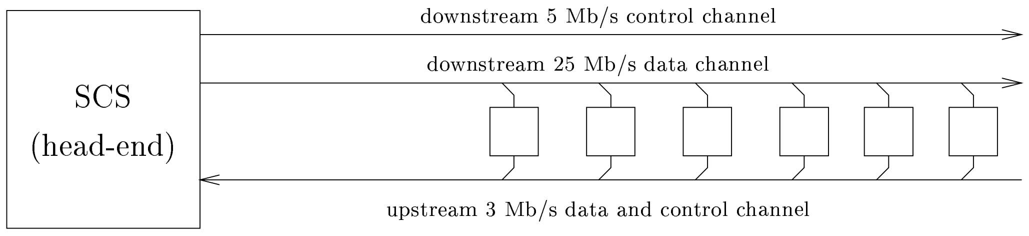

downstream 5 Mb/s control channel
downstream 25 Mb/s data channel
SCS
(head-end)
upstream 3 Mb/s data and control channel

Figure 1: MXL Network Topology

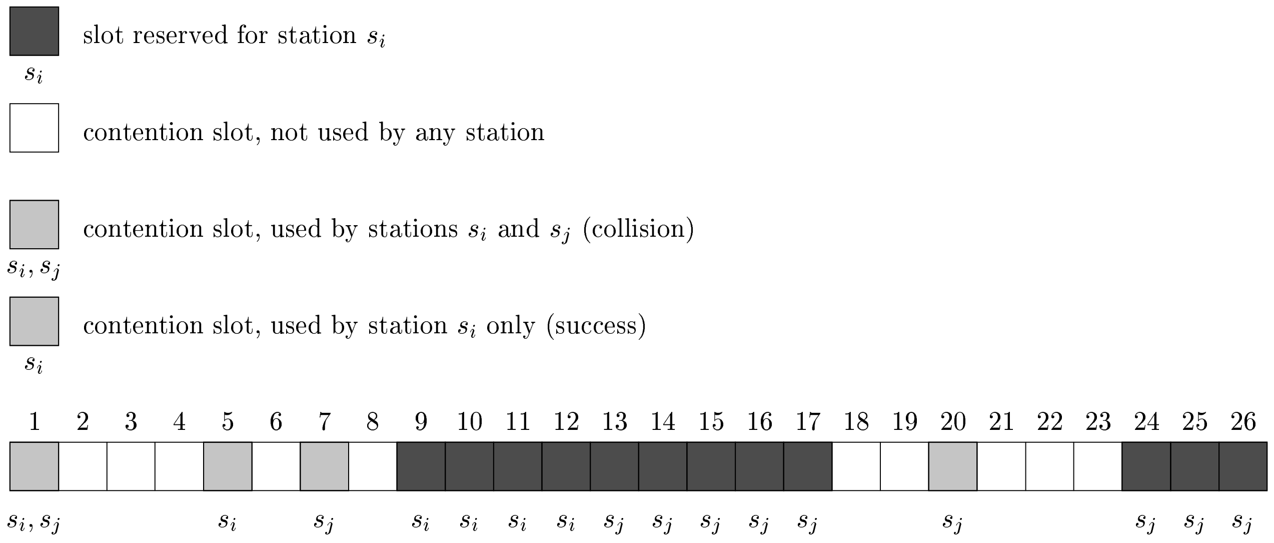
In order to access the upstream channel, a CM needs to send the SCS a reservation request. To this end, the CM waits for a contention slot on the upstream channel and transmits a short reservation control packet which specifies the CM's identity and the number of slots needed for successive transmission of all its waiting data packets. The MXL SCS always returns the acknowledgment for an upstream reservation packet in the next slot on the downstream control channel. The period of time elapsed between transmitting a successful reservation and receiving an acknowledgment is therefore fixed, and will be referred to as an "ACK-window". If a station does not get an acknowledgment during the ACK-window, it assumes a collision has occurred. It then waits some period of time, determined according to the contention algorithm, and re-contends using another contention slot.

A short upstream data packet, that fits into a slot, can be transmitted during a contention time slot with no preceding reservation. Though it contains no reservation, such a packet needs to be acknowledged by the SCS during the ACK-window. In contrast, packets transmitted in reserved slots do not have to be acknowledged by the SCS since they cannot collide with other packets. Such packets can still be lost due to transmission errors, but recovering from such losses is the role of upper layer protocols (like TCP).

During the sign-on process, each station empirically determines the number of slots in the ACK-window in the following way. Upon being powered-on, the station waits for an upstream contention slot and transmits a sign-on request control packet to the SCS. It then waits for an acknowledgment from the SCS. If an acknowledgment is not received within one millisecond (an upper bound on the ACK-window) the station waits for another contention slot and retransmits its sign-on request. If an acknowledgment is received, the station considers the period of time between the transmission of the last request and the reception of the acknowledgment as its ACK-window. Note that due to the way the upstream channel is slotted, all the stations have the same ACK-window regardless of their distance from the SCS.

An execution example of the MXL protocol is given in Figure 2, assuming that the ACK-window is 3 slots. This means that a station that transmits its reservation request successfully (i.e. with no collision) on slot i of the upstream channel will get an acknowledgment on the downstream control channel when slot number $i+3$ is scheduled on the upstream channel. In the depicted example there are two active stations: s_i and s_j. Both stations transmit during slot 1. Thus, a collision occurs and neither of them get an acknowledgment during slot 4. According to some contention algorithm, like ALOHA or the tree algorithm (see [7] for more details), station s_i tries again, in slot 5 say, and succeeds. Thus, it gets an acknowledgment from the SCS on the downstream control channel when slot 8 is scheduled on the upstream channel. The acknowledgment packet has a field called *Reservation_Delay_Slot_Count,* which tells the contending station the number of upstream slots after which the station can start transmitting its allocated quota. In this particular case the SCS has no pending reservation when the request from s_i is received. Thus, assuming that s_i requests 4 slots, the SCS grants s_i 4 slots starting immediately after s_i receives the acknowledgment. Hence the value of the *Reservation_Delay_Slot_Count* field is 0, and slots 8-11 are used by s_i with no interference.

Station s_j transmits its reservation for 5 slots during upstream slot 7. Since there is no collision, s_j receives an acknowledgment over the downstream control channel during time slot 10 of the upstream channel. This acknowledgment informs s_j that it is granted the requested number of slots. However, the *Reservation_Delay_Slot_Count* field is 2 rather than 0 because the next couple of slots are in the middle of a burst reserved for s_i. After station s_j finishes transmitting its 5-slot packet(s), in slot 17, the channel is again available for



slot reserved for station s_i
s_i
contention slot, not used by any station
contention slot, used by stations s_i and s_j (collision)
s_i, s_j
contention slot, used by station s_i only (success)
s_i
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
s_i, s_j s_i s_j s_i s_i s_i s_i s_j s_j s_j s_j s_j s_j s_j s_j s_j

Figure 2: MXL Execution Example

contention.

Unlike other schemes (e.g. [4, 8]), MXL does not allow a transmitting station to use the reserved slots in order to send another request. New requests can be sent using contention slots only. This guarantees fairness among stations without complicating the algorithms performed by the SCS or the algorithm performed by the stations.

The MXL protocol, as described so far, divides the upstream channel into two logical channels: a contention channel which consists of all the available (contention) upstream slots, and a reserved channel which consists of all the reserved slots. The total throughput S of the upstream channel can be expressed as

$$S = \frac{\sigma}{\sigma + 1/S'}$$

where σ is the average number of slots in each reservation request and S' is the throughput of the contention channel.

Like MXL, most of the architectures for HFC networks use the concept of separating the upstream channel into a contention logical channel and a reserved logical channel, and employing a contention resolution algorithm in the contention channel. However, the MXL is distinguishable from other architectures by its following properties:

- A packet is transmitted as one unit, using a burst of reserved slots. Thus, fragmentation of packets at the stations and re-assembly at the SCS are avoided.
- A side-effect of the previous property is that the SCS allocation mechanism is simple. The SCS needs to maintain only a local counter whose value indicates the offset to the

next available slot. When a request for ρ slots is accepted, the value of the pointer is returned to the requesting station, and the counter is incremented by ρ. This simple allocation method reduces the SCS processing time, and therefore the ACK-window duration, to a minimum. Consequently, stations are informed as early as possible of the results of their last contention, and the throughput of the channel increases.

- By preventing stations from sending new reservation requests in the reserved channel starvation is avoided, and fairness is achieved to some extent, without complicating the SCS allocation algorithm.

A study of possible contention algorithms for MXL with performance comparison is presented in [7].

3 Supporting Synchronous Traffic

3.1 Synchronous Traffic

In order to support hard real time applications like voice and video that arrive at regular intervals and require timely delivery, or in order to support home users who are willing to pay more in order to get a guaranteed bit-rate for their applications, the link layer of a CM network should be able to provide to its customers a guaranteed amount of the channel bandwidth with a bounded time between successive transmissions. This kind of service is also required from the link layer by some higher layer protocols, like TENET[1], that supports real-time traffic generated by the transport layer.

The first protocol to support such service requirements was the MAC protocol of the FDDI. FDDI uses the timed token protocol [6], which distinguishes between *synchronous* and *asynchronous* packets. Synchronous packets, such as packetized voice or video, are generated at regular intervals and have delivery time constraints. Asynchronous packets, in contrast, are non-periodic and have no time constraints. The time token protocol guarantees to each source of a synchronous traffic an average bandwidth and a bounded delivery time. The latter depends on some parameters, mainly the TTRT [6], selected during ring initialization.

Supporting synchronous traffic on the downstream channel is relatively simple. This is because this channel is governed by a single entity, the SCS, that can schedule packets for downstream transmission according to any pre-determined set of rules. In contrast, supporting upstream synchronous traffic is a difficult challenge, because of the distributed access to the upstream channel. Therefore, this paper concentrates upon synchronous traffic on the upstream channel only.

Cable-modem networks that support synchronous traffic on the upstream channel, like [2, 4, 8], require that the *asynchronous* packets will be broken down into a stream of fixed sized units (ATM or non-ATM cells) at the sending CM, and will be re-assembled at the SCS after being delivered. As a typical example, consider the cable-modem network described in [4]. The SCS divides the time domain into a series of successive frames. Each frame is divided into synchronous and asynchronous regions, each having multiple time slots. The boundary between the two regions can be changed dynamically in order to accommodate set up of new synchronous calls or take down of existing ones. A synchronous call that acquires

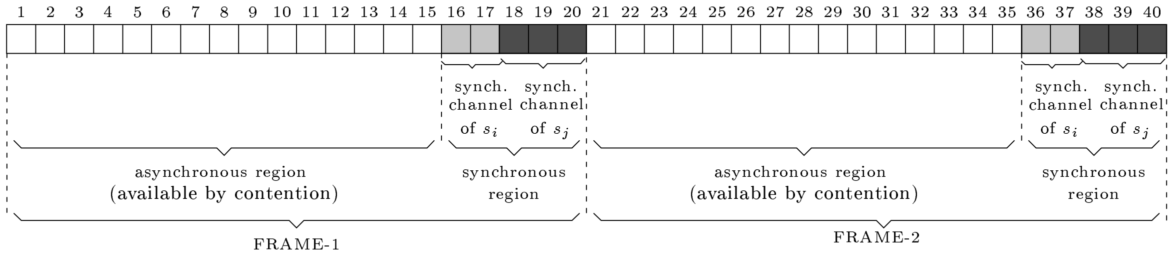


Figure 3: A Naive Approach for Synchronous Traffic Support

synchronous bandwidth makes use of a periodic time slot allocation in the synchronous region of each frame.

However, if segmentation of the asynchronous packets is to be avoided, as in the MXL CM network, the support of synchronous traffic becomes a greater challenge. A naive solution to this problem is as follows.

3.2 Ignoring Reservations

As in [4], the SCS divides the time domain into a series of successive frames, and every frame is divided into synchronous and asynchronous regions (see Figure 3). Again, each region has multiple time slots, and the boundary between the two regions can be changed by the SCS dynamically, according to the synchronous traffic load. If the SCS receives a reservation requests for an asynchronous packet, that it cannot accommodate in the asynchronous region of the current frame, it ignores the request. This idea is depicted in Figure 3. In this figure it is assumed that the frame length is $\tau_f = 20$ slots, and that each frame has $\tau_a = 15$ slots in the asynchronous region and $\tau_s = 5$ slots in the synchronous region. The values of τ_a and τ_s can be pre-negotiated with the synchronous sources in order to guarantee the highest quality of service constraints. In Figure 3 it is assumed that there exist two synchronous sources: the first is station s_i that has acquired 2 slots in every synchronous region, and the second is station s_j that has acquired 3 slots in every synchronous region. The request for synchronous bandwidth is made by a CM by means of a call set-up protocol. To this end, the CM exchanges asynchronous call set-up packets with the SCS. The allocation is guaranteed until the station executes a call take-down protocol, informing the SCS that the synchronous bandwidth is no more needed. In terms of the example in Figure 3, this means that the allocation of 2 and 3 slots in the synchronous region of each frame is guaranteed to s_i and s_j as long as needed, without the need to re-contend for this allocation.

To see why the SCS needs to ignore reservation requests, suppose that a reservation for 10 slots is made by station s_k in slot 5 of FRAME-1. Suppose also that the ACK-window, namely the period of time elapsed between transmitting a reservation that does not collide with other reservations and receiving an acknowledgment, is $\delta = 3$. Hence, the reserving station s_k can start transmitting only in slot number 9. But s_k must stop transmitting after slot 15, because slot 16 is pre-allocated for the synchronous traffic of s_i. If segmentation of asynchronous packets is to be avoided, the reservation of s_i cannot be accommodated in slot 9.

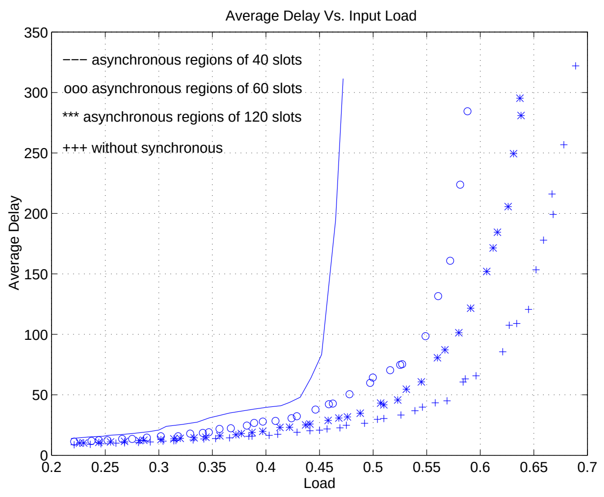


Figure 4: The Performance of the Asynchronous Channel When the SCS needs to Ignore Successful Reservations

Ignoring a reservation that cannot be fully accommodated in the asynchronous region of the current frame is a straightforward solution that keeps the SCS algorithm simple. In such a case the SCS does not send an acknowledgment to the requesting station, and the latter will re-contend in a future asynchronous contention slot according to the rules of the contention algorithm, exactly as if a collision has occured. However, this approach has a significant negative effect on the performance of the *asynchronous* upstream channel, as shown in Figure 4. The graph depicts the average access delay to the asynchronous upstream channel, normalized to the slot transmission time, versus input load[2] for the cases where $\tau_a = 40$ slots, $\tau_a = 60$ slots and $\tau_a = 120$ slots. The graph depicts also the case where synchronous traffic is not accommodated ($\tau_a \to \infty$), and therefore the SCS does not have to ignore reservations. The results have been achieved by simulating 128 stations, assuming that ALOHA is used in conjunction with the binary exponential back-off algorithm in order to resolve collisions. It is also assumed that a slot is 64-byte long, and that 30.4% of the packets are 2-slot long, 8.3% of the packets are 3-slot long, 8% of the packets are 4-slot long, 10% of the packets are 10-slot long, 25% of the packets are 18-slot long, and 18.3% of the packets are 24-slot long. Hence, the average packet length is 11 slots. This packet length distribution, determined for IP traffic based on [5], does not necessarily reflect the precise traffic of the future home users. However, the illustrated problem takes place for any reasonable packet length distribution. The graph shows that for $\tau_a = 40$, throughput is

[2]The input load is normalized to the bandwidth available for the asynchronous traffic. Thus, the performance differences are attributed to the way the asynchronous bandwidth is used when synchronous traffic is accommodated, rather than to the amount of bandwidth left for the asynchronous traffic due to the synchronous traffic requirements.

reduced by more than 30% from 0.69 (69% of the upstream transmission speed) to 0.47 and the average delay is substantially higher for every input load. The throughput loss happens due to the following reasons:

1. Successful transmissions in contention slots are ignored by the SCS, and are therefore considered as collisions by the transmitting stations.
2. Sequences of slots that are not long enough to accommodate the reservations received by the SCS remain empty.

Obviously, as τ_a increases, the negative effect of the synchronous traffic is reduced. This is because the probability that the SCS will have to ignore a successful reservation gets smaller. However, increasing τ_a leads to the increase of the delay between successive transmissions of *synchronous* data. In order to support real-time voice or video, the "local loop" delay should be smaller than 10ms, which is translated to $\tau_f < 59$ for an upstream rate of 3Mb/s.

3.3 Using Double ACKs

In order to achieve better performance, the SCS should not ignore reservation requests. A possible solution might be to use double acknowledgments, as proposed in [8], in the following way. When the SCS receives a reservation, it sends an "empty" ACK to the reserving CM. Such an ACK indicates that the reservation has been recorded and will be accommodated in the future. Later, when the SCS finds enough consecutive slots for the reservation, it allocates them to the reserving station and sends another acknowledgment. Unlike the first acknowledgment, the second one contains an offset to the upstream slot where the transmission should begin.

The performance of this approach depends on the exact scheduling algorithm used by the SCS. This algorithm should be designed based on the fairness policy and some other relevant factors. For instance, if the SCS has an old pending reservation for 30 slots and two newer pending reservations for 20 slots each, then when a new frame is scheduled for which τ_a=40 slots, the SCS can decide either to accommodate the old reservation and to waste 10 slots, or to accommodate the two newer reservations in order to achieve better performance while sacrificing fairness. However, even the most sophisticated algorithm cannot prevent the loss of a sequence of asynchronous slots which is too short to accommodate the shortest pending reservation (e.g., in the above example if the SCS had only 10 available asynchronous slots). This means that this scheme eliminates the first cause to throughput loss in the scheme of Section 3.2, but not the second one.

Another drawback of this approach is that it complicates the SCS and the CM implementation. If the second acknowledgment is lost, an event which is very likely in HFC networks due to their susceptibility to a variety of radio-frequency impairment, the reserving CM will keep waiting for a long time. In order to avoid dead-lock, the CMs can use a time-out after which the last reservation request will be re-transmitted. But a safer time-out value, which will not expire prematurely, may increases the average delay significantly.

4 The Proposed Scheme

The following section presents a new scheme for supporting the transmission of synchronous traffic in MXL, while avoiding the drawbacks associated with the solutions described in Section 3. The proposed scheme has the following properties:

- All the advantages of the MXL, as presented at the end of Section 2 prevail. Namely:
  - asynchronous packets do not have to be fragmented before being transmitted;
  - the algorithm employed by the SCS for asynchronous bandwidth allocation is simple, and can be implemented using only a local counter whose value indicates the offset to the next available slot;
  - fairness among asynchronous sources is guaranteed.
- When $\delta = 0$, asynchronous slots are not wasted at all due to the synchronous traffic. If δ is larger than 0, the wastage is minimal.

4.1 The Guaranteed Quality of Service

The properties mentioned above are achieved while providing the synchronous sources the same quality of service provided by the FDDI, which is a relaxation of the quality of service provided in the schemes of Section 3. In FDDI a synchronous source gets a synchronous sub-channel that enables the source to transmit its synchronous data periodically. The delay between consecutive transmissions is variable, rather than fixed. However, this delay has a pre-determined average (of $1 \cdot \text{TTRT}$) and a pre-determined upper bound (of $2 \cdot \text{TTRT}$). Since the synchronous source can transmit a fixed amount of data during each access to its synchronous sub-channel, each sub-channel has in fact a guaranteed bandwidth.

The proposed scheme allocates to each synchronous source a synchronous sub-channel with a guaranteed bandwidth a guaranteed average delay $T_{average}$ between consecutive transmissions and a guaranteed maximum delay $T_{\max}$ between consecutive transmissions. The value of $T_{average}$ can be as short as needed, while $T_{max} = T_{average} + \gamma$ where γ is equal to the longest asynchronous burst a station can transmit following a single reservation. For instance, suppose that every contending station is allowed to reserve bandwidth for only one asynchronous packet during each reservation request. Suppose also that the longest packet is 1518 bytes (as in Ethernet), namely 24 slots in a 64-byte slotted MXL. Thus, for a 3Mb/s upstream channel $T_{max} = T_{average} + 24 * 64 * 8/(3 * 10^6)$seconds.

As in the schemes described in Section 3, the SCS divides the time domain into a series of successive frames, and each frame is divided into two regions: one for synchronous traffic and another for asynchronous traffic. The average length of each region depends on the percentage of bandwidth needed to be assigned for each class of traffic and on the desired value of $T_{average}$. Let $\tau_f = \tau_s + \tau_a$ slots be the average length of each frame, where τ_s slots are for synchronous traffic and the remaining τ_a slots are for asynchronous traffic. This implies that $T_{average} = \tau_f$, that τ_s/τ_f of the bandwidth is allocated to synchronous traffic, and that the remaining τ_a/τ_f is allocated to asynchronous traffic. The values of τ_f, τ_s and τ_a can be adjusted by the SCS from time to time in order to accommodate new set-up and tear-down requests of synchronous calls, or in order to change the quality of service. For instance, by decreasing τ_s and τ_a by the same factor the SCS can support synchronous calls that need smaller average delay between successive accesses, without changing the total bandwidth assigned to each traffic class.

4.2 Frame Extension

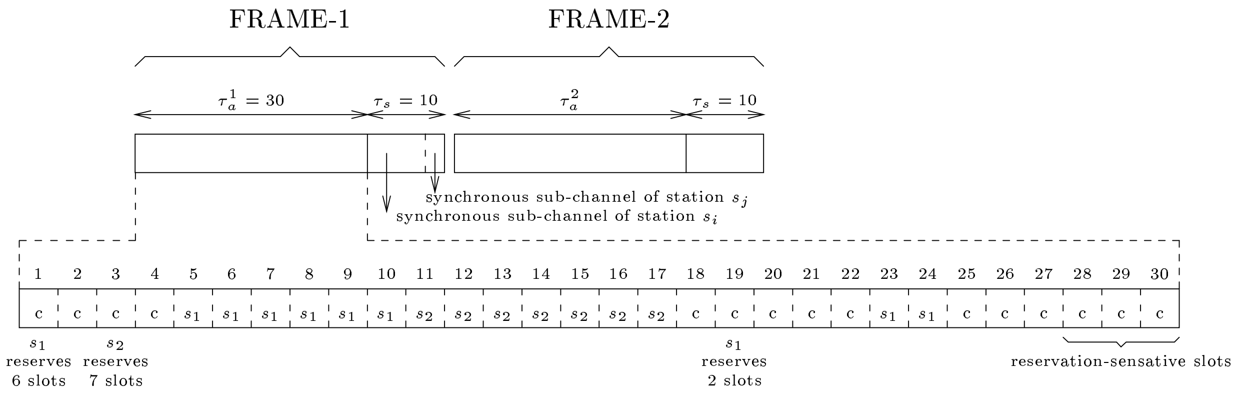
In order to avoid loss of asynchronous bandwidth when the number of asynchronous slots remaining in some frame is less than the number needed for the transmission of an asynchronous packet by a reserving station, the SCS is allowed to extend the asynchronous part of a frame. However, in order to keep the average size of a frame fixed, thus guaranteeing the value of $T_{average}$, the SCS will have to reduce the size of the asynchronous region in subsequent frame(s) by the same number of slots. For instance, suppose that a station makes a reservation for λ slots in order to transmit its asynchronous packet. Suppose also that this reservation is made when only $\lambda' < \lambda$ slots are left in the asynchronous region of the frame where the reservation is received. Instead of ignoring the reservation, the SCS extends the asynchronous region by $\lambda - \lambda'$ slots. This causes a delay of $\lambda - \lambda'$ to the slots of the synchronous region in the same frame. In order to guarantee the value of $T_{average}$, the SCS tries to remove the extra $\lambda - \lambda'$ slots from the asynchronous region of the subsequent frame. If this is not possible, because of reservations received in that frame or because $\lambda - \lambda' > \tau_a$, the SCS removes the overdraft from the earlier subsequent frames.

An important property of the proposed scheme is that it needs to be executed only by the SCS. The cable-modem stations do not have to know about the length of each frame, about the extension of a frame, or when the channel is switched from an asynchronous region to a synchronous region and vice versa. In particular, cable-modems that do not implement the synchronous scheme can work in the same MXL network with cable-modems that do implement it.

4.3 An Example of the New Scheme

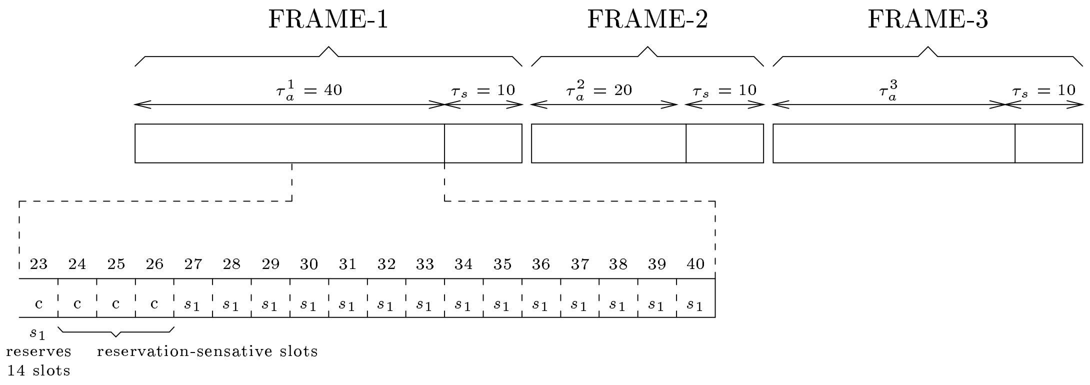
A formal description of the scheme is presented in Section 6. In the following we give a detailed description by means of an example. Figure 5 demonstrates the scheme in three different scenarios. It is assumed that the average length of a frame is $\tau_f = 40$ slots, the average length of the asynchronous region is $\tau_a = 30$ slots, and the fixed length of the synchronous region is $\tau_s = 10$ slots. It is also assumed that the synchronous channel is divided between two stations, s_i and s_j, such that 8 slots are allocated to s_i and 2 to s_j. Assuming that the upstream transmission rate is 3 Mb/s and that the slot size is 64 bytes, the synchronous channel of s_i has 600 Kb/s, the synchronous channel of s_j has 150 Kb/s, and the average delay between subsequent synchronous transmissions is 40 slots = 6.82 ms. Recall that the maximum delay between subsequent synchronous transmissions, T_{max}, is larger than the average delay $T_{average}$, by the transmission time of the longest asynchronous burst a station can transmit following a single reservation.

Since the length of the asynchronous region of a frame is not fixed, τ_a indicates the default size, which is also the average one. The actual length of this region in a specific frame, FRAME-i say, is represented by τ_a^i. The length of the synchronous region of a frame can be either fixed or variable, depending on the network policy. If a CM must be assigned the same number of synchronous slots in every frame, then the size of the synchronous region is fixed (as long as new synchronous calls are not set up and existing ones are not taken

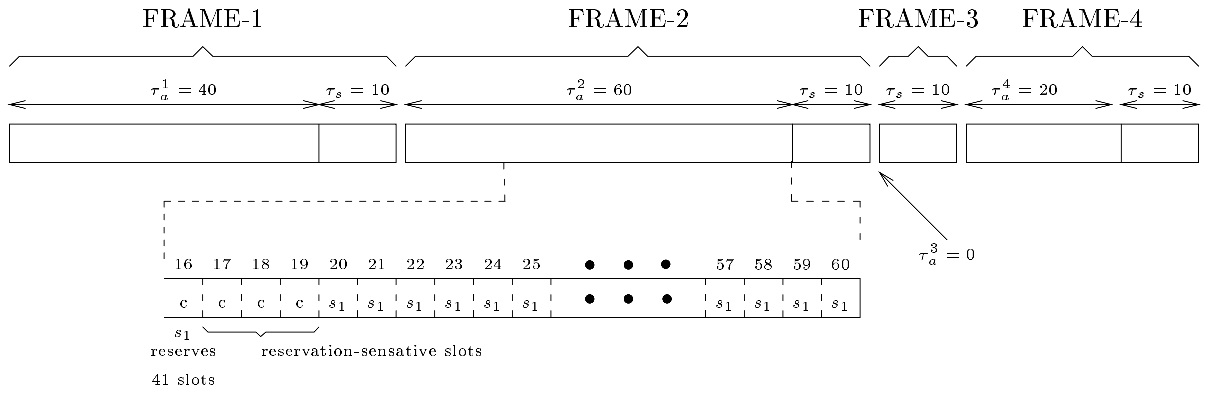


FRAME-1
FRAME-2
$\tau_a^1 = 30$
$\tau_s = 10$
τ_a^2
$\tau_s = 10$
synchronous sub-channel of station s_j
synchronous sub-channel of station s_i
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
c c c c s_1 s_1 s_1 s_1 s_1 s_1 s_2 s_2 s_2 s_2 s_2 s_2 s_2 c c c c c s_1 s_1 c c c c c c
s_1 reserves 6 slots
s_2 reserves 7 slots
s_1 reserves 2 slots
reservation-sensative slots

(a) the SCS does not need to extend the asynchronous region of FRAME-1



(b) the SCS extends the asynchronous region of FRAME-1 and reduces it in FRAME-2



(c) the SCS extends the asynchronous region in FRAME-1 and FRAME-2 and remove the asynchronous region in FRAME-3

Figure 5: Three Execution Examples of the Proposed Scheme

down). However, a more flexible allocation method will allow the SCS to save synchronous bandwidth by assigning to a CM a different number of synchronous slots in each frame. For instance, a CM that needs a channel of 30Kb/s will be assigned one synchronous slot every two frames, which is equivalent to 37.5Kb/s, instead of one slot every one frame (75Kb/s). However, since this issue is orthogonal to the proposed scheme, we will assume for simplicity that the length of the synchronous region of a frame is fixed. Hence, we shall use τ^i to represent both the average length of the synchronous region and the actual length in a specific frame. Since the asynchronous region has a variable length, the whole frame as a variable length as well. Thus, the average size of the whole frame is represented by τ_f, whereas the actual size of FRAME-i is represented by τ_f^i.

Consider first Figure 5(a). This figure shows the case where the SCS does not need to extend the length of the asynchronous region of FRAME-1. Hence, $\tau_f^1 = \tau_f = 30$ holds. The figure shows the exact contents of the asynchronous slots in the FRAME-1. A slots marked as 'c' is available for contention, whereas a slot marked as 's_i' is reserved for station s_i. Station s_1 reserves 6 slots in slot 1. Assuming that the ACK-window is of $\delta = 3$ slots, s_1 gets an acknowledgment is slot 4 and starts transmitting in slot 5. Station s_2 sends a reservation in slot 3, and it starts transmitting its asynchronous packet in slot 11, immediately after s_1 finishes. The last reservation for asynchronous transmission is made during slot 19 by station s_1 that reserves 2 slots and is allocated slots 23 and 24. Since no reservation is made later, the SCS can assign the 10 slots following slot 30 of the asynchronous region to the synchronous region of FRAME-1. Since the values of τ_f^i and τ_a^i can dynamically change, the SCS must inform s_i and s_j before every synchronous transmission. Hence, the SCS sends an 'acknowledgment' to stations s_i, informing this station that it can transmit its synchronous data during slots 31-38, and another acknowledgment to station s_j, informing this station that it can send its synchronous data in slots 39-40. These acknowledgments are similar to those sent to stations that reserve asynchronous bandwidth, but with one important difference: there is no need for a synchronous station (s_i and s_j in our example) to send an explicit reservation request for synchronous bandwidth every frame. Only one request is made, by means of an upper (Network) layer call set-up protocol like Q.931 [3]. This reservation prevails until the call is taken down, again by means of an upper layer protocol.

Next, consider the example in Figure 5(b). This example demonstrates the case where the asynchronous region of FRAME-1 is extended by 10 slots, which are then removed from the asynchronous region of FRAME-2. In slot 23 of the asynchronous region of FRAME-1, station s_1 reserves 14 slots. Since $\delta = 3$, the earliest time when s_1 can start transmitting its asynchronous packet is slot 27. In order to accommodate the whole packet, the asynchronous region of FRAME-1 should be extended by 10 slots. Thus, the asynchronous transmission of s_i is scheduled to slot 41 and the asynchronous transmission of s_j is scheduled to slot 49. This implies that the delay between the synchronous transmission in FRAME-1 and the synchronous transmission in the previous frame is larger by 10 slots than the guaranteed average delay of $\tau_f = 40$. In order to compensate for the 10 "stolen" asynchronous slots in FRAME-1, the *planned* size of the asynchronous region in FRAME-2 is $\tau_a - 10 = 20$ slots. Assuming that the SCS is not required to extend the length of this region (the exact contents of the slots in the asynchronous region of FRAME-2 is not shown in Figure 5(b)), τ_a^2 is indeed 20. Thus, the delay between the synchronous transmission in FRAME-2 and the synchronous transmission in the previous frame is smaller by 10 slots than the guaranteed average delay of $\tau_f = 40$, and the average delay of the last two accesses is exactly $(50 + 30)/2 = 40 = \tau_f$ slots.

Finally, consider Figure 5(c). Suppose that the contents of the asynchronous slots in FRAME-1 is the same as in the previous case (Figure 5(b)). Thus, the planned size of the asynchronous region in the second frame is 20 slots. The figure shows the exact contents of the asynchronous region of FRAME-2 starting in slot 16. In slot 16 station s_1 makes a reservation of 41 slots. In order to accommodate this reservation, the SCS needs to extend the length of the asynchronous region to 60. Consequently, the asynchronous channel has an aggregate overdraft of 40 slots. Since this overdraft is larger than τ_a, FRAME-3 has no asynchronous region (i.e. $\tau_a^3 = 0$). This reduces the overdraft to only 10 slots, so the planned length of the asynchronous region of FRAME-4 is 20. Assuming that the SCS does not need to extend this asynchronous region (e.g. because the reservation pattern is similar to the one received in FRAME-2 of Figure 5(b)), τ_a^4 is indeed 20, and the average delay of the last four accesses of s_i and s_j to the synchronous channel is exactly $(50 + 70 + 10 + 30)/4 = 40 = \tau_f$ slots.

5 "Reservation-Sensitive" Slots

Despite the SCS's ability to extend the length of an asynchronous region when necessary, there are some cases where due to the non-zero delay between the SCS and the CMs, reservation requests for asynchronous bandwidth will have to be ignored by the SCS or deferred to a future frame. There are two reasons for this, both depicted in Figure 5. Consider the execution example in Figure 5(a) first. Since $\delta = 3$ slots, when upstream slot 27 is scheduled, the SCS should determine the status of slot 31 and notify the stations whether it is reserved (either for asynchronous or synchronous transmission) or it is available for contention. Since there is no pending reservation, the SCS determines in this case that slot 30 will be the last one of the asynchronous region and that the following 8 slots (31-38) will be assigned to station s_i for synchronous transmission. The SCS transmits on the downstream control channel a short control packet that notifies all the stations that the next upstream slot (31) is reserved. Another short control packet is sent by the SCS to inform s_i that it can transmit its synchronous data using the next 8 upstream slots. This implies that any reservation made by the stations in the last δ slots of the synchronous region of FRAME-1, after τ_f^1 has already been determined, cannot be accommodated in this frame.

Before discussing the other case where a reservation needs to be ignored by the SCS, note that T_{max} — the maximum delay between two successive accesses of a synchronous source to the upstream channel — is equal to the maximum possible value of τ_f^i, that is $\tau_a + \tau_s + \tau_e$, where τ_e is the longest possible extension of the asynchronous region. Since τ_a and τ_s are fixed, T_{max} can be minimized only by minimizing τ_e. Recall that γ is the longest asynchronous burst a station can reserve in a single reservation request. This implies that $\tau_e \geq \gamma$. For instance, $\tau_e = \gamma$ if a successful reservation for γ slots is made in slot 27 of FRAME-1 in Figure 5(a).

Consider now Figure 5(b). Note that slots 24-26 of FRAME-1 in this example are available for contention and that station s_1 makes a successful reservation in slot 23. Due to this reservation, the SCS needs to extend the length of the asynchronous region by 10 slots to 40. If, however, the SCS receives and accommodates reservations during slots 24-26, the length of the asynchronous region will have to be extended by at most $\delta \cdot \gamma$ slots in addition to the extension made due to the reservation of s_1 in slot 23. Consequently, the maximum value of τ_e becomes $(\delta + 1)\gamma - \delta$, which might be too high for many synchronous applications, like voice and video. Therefore, in order to guarantee that τ_e does not exceed its minimum value of γ, reservations that are received by the SCS during the δ slots *after the asynchronous region of a frame was extended beyond its planned length* should not be accommodated in the same frame.

In the following, the slots during which received reservations cannot be accommodated in the same frame due to one of the two reasons indicated above will be referred to as *"reservation-sensitive" slots.* Note that in any frame there are no more than δ reservation-sensitive slots. If the asynchronous region is extended beyond its planned length then there exist δ reservation-sensitive slots after the slot with the last reservation. If the asynchronous region is not extended beyond its default length, then the last δ slots or a portion thereof can be reservation-sensitive.

The straightforward approach to handle the reservations received during reservation-sensitive slots is to ignore them. This means that the SCS will not send an acknowledgment to a station whose reservation is received in a reservation-sensitive slot, and the station will re-contend in a future contention slot in some succeeding frame, exactly as if a collision has occured. In the worst case, this approach wastes a fraction of $p \cdot \delta/\tau_a$ from the asynchronous bandwidth, where p is the probability for a successful reservation in a contention slot. In most combinations of network topology and quality of service required, $\delta/\tau_a < 0.1$ and $p < 0.3$ hold. Hence, the wasted bandwidth is small. Figure 6 depicts the average access delay to the asynchronous upstream channel versus input load for $\tau_a = 40$ and $\delta = 4$ for three cases: when the new scheme is employed, when the scheme from Section 3.2 (the "naive approach") is employed and when synchronous traffic is not employed.

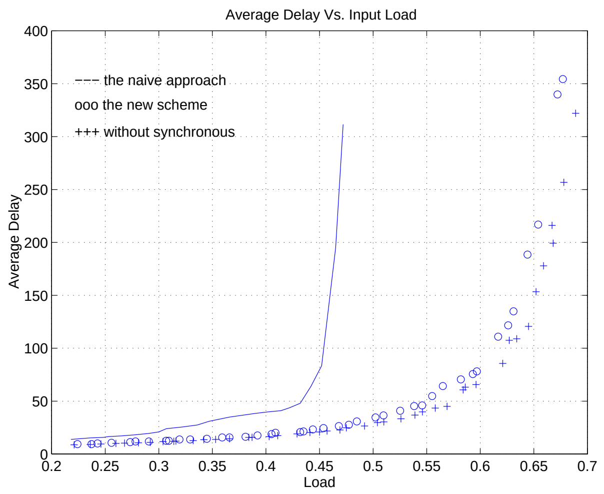


Figure 6: The Performance of the Asynchronous Channel When the New Scheme is Employed

The results in Figure 6 are effective for $\tau_a = 40$ and $\delta = 4$, regardless of the exact values of τ_f and τ_s. This is because as in the case of Figure 4 the input load is normalized to the bandwidth available for the asynchronous traffic. The results have been obtained for the same conditions described in Section 3.2 with regard to the graph in Figure 4. It is evident from Figure 6 that under the proposed scheme the effect of the synchronous traffic on the performance of the asynchronous channel is negligible. The reason that a small difference does exist is attributed to the reservation-sensitive slots. Hence, this small difference vanishes as $\delta/\tau_f \to 0$ (e.g. when $\delta = 0$).

In an HFC network with a long distance between the SCS and the cable-modems the value of δ might be too large. If it is not possible to keep δ/τ_a low by increasing τ_a then the following approach can be used. Instead of ignoring all the reservations received during the reservation-sensitive slots, the SCS will accept and accommodate the first reservation and ignore the others. This will reduce the wasted bandwidth to max(0,$(p \cdot \delta - 1)/\tau_a$). Since the accepted reservation cannot be accommodated in the same frame, it will be accommodated in the first slot of the next frame with an asynchronous region. In general, accommodating reservations in future frames prevents the SCS from changing the values of τ_a and τ_s, and therefore from setting up or taking down synchronous calls, until every pending reservation is accommodated. If, however, only one reservation can be transfered to a future frame, the set-up and take-down of synchronous calls are not affected.

6 Formal Description and Proof of Correctness

In this section we give a formal description of the proposed scheme, and prove that it indeed guarantees a bounded maximum delay, an average delay, and dedicated bandwidth to each synchronous station. The proofs appear in Theorem 1, 2 and 3 respectively. Before presenting the proofs, the following is a summary of the relevant notations:

$\boldsymbol{T_{max}}$, $\boldsymbol{T_{average}}$ – The maximum delay and the average delay between two consecutive accesses of a synchronous source to the upstream channel.

$\boldsymbol{\tau_f}$, $\boldsymbol{\tau_a}$, $\boldsymbol{\tau_s}$ – The default sizes of a frame, the default size of the asynchronous region of a frame, and the fixed size of the synchronous region of a frame.

$\boldsymbol{\tau_f^i}$, $\boldsymbol{\tau_a^i}$ – The actual size of FRAME-i and the actual size of the asynchronous region of FRAME-i.

$\boldsymbol{\alpha^n}$ – The aggregated overdraft of the asynchronous regions in FRAME-1$\cdots$FRAME-n $\sum_{i=1}^{n} \tau_a^i - n \cdot \tau_a$ (this equality follows by the SCS algorithm as presented later).

$\boldsymbol{\hat{\tau_a}^i}$ – The *planned* size of the asynchronous region of FRAME-i, determined by the SCS after accounting for the previous aggregated overdraft.

$\boldsymbol{\gamma}$ – The longest asynchronous burst a single station can transmit following a single reservation.

$\boldsymbol{\delta}$ – The length of the ACK-window.

A formal description of the scheme is presented in Figure 7. Note that the algorithm does not allow to extend $\hat{\tau_a}^i$, the planned length of FRAME-i, by more than γ slots. However, it does not dictate a specific approach for handling reservations received during the reservation-sensitive slots.

```
1 let FRAME-1 be the first frame since the last time the SCS has determined the values of
    $\tau_f$, $\tau_a$ and $\tau_s$, and define $\alpha^0$ to be 0
2 before FRAME-$i$ is scheduled, set $\hat{\tau_a}^i \leftarrow \tau_a - \alpha^{i-1}$
3 if $\hat{\tau_a}^i \leq 0$, then:
    3.1 $\tau_a^i \leftarrow 0$, which means that FRAME-$i$ will have no asynchronous region
    3.2 $\alpha^i \leftarrow \alpha^{i-1} - \tau_a$
4 else (i.e., $\hat{\tau_a}^i > 0$):
    4.1 FRAME-$i$ has an asynchronous region, which consists of two parts: an original
        part which consists of the first $\hat{\tau_a}^i$ slots, and an extended part which consists of
        the remaining 0 or more slots
    4.2 if all the reservations received during the first $\hat{\tau_a}^i - \delta$ slots of the original part a
        reservation can be accommodated in the original part, then:
        4.2.1 the extended part does not exist, namely $\tau_a^i = \hat{\tau_a}^i$
        4.2.2 $\alpha^i \leftarrow 0$
    4.3 else, namely a reservation which cannot be accommodated in the original part is
        received during the first $\hat{\tau_a}^i - \delta$ slots of the original part:
        4.3.1 FRAME-$i$ is extended by $e$ slots in order to accommodate the received
              reservation.
        4.3.2 accommodate no more reservations in FRAME-$i$; hence $\tau_a^i = \hat{\tau_a}^i + e$
        4.3.2 $\alpha^i \leftarrow e$
```

Figure 7: A Formal Description of the Algorithm Performed by the SCS

Theorem 1
$T_{\max} \leq \tau_f + \gamma$

Proof
Consider a synchronous source transmitting in FRAME-$(i-1)$ and in FRAME-i. The total delay between the two accesses of this source to the synchronous channel is $\tau_s + \tau_a^i$. Since $\tau_f = \tau_s + \tau_a$, it remains to show that $\tau_a^i \leq \tau_a + \gamma$. By the SCS algorithm (Figure 7), the original part of FRAME-i has $\hat{\tau_a}^i = \tau_a - \alpha^{i-1}$ slots and the extended part has no more than γ slots. Thus, $\tau_a^i \leq \hat{\tau_a}^i + \gamma \leq \tau_a - \alpha^{i-1} + \gamma \leq \tau_a + \gamma$, and the theorem holds. □

Corollary 1
In order to minimize $T_{\max}$ while avoiding the need to break down asynchronous packets, γ should be set to the number of slots in the longest asynchronous packet. □

Theorem 2
For a synchronous call lasting n frames, $T_{\text{average}} = \tau_f + \gamma/(n-1)$. Hence, if the call is "sufficiently long", such that $\gamma/(n-1) \to 0$, $T_{\text{average}} = \tau_f$.

Proof
Let the first frame where the synchronous call is accommodated be FRAME-1 and[3] the last one be FRAME-n. For every i, $2 \le i \le n$, the delay between the transmission in the synchronous region of FRAME-$(i-1)$ and the transmission in the synchronous region of FRAME-i is $\tau_s + \tau_a^i$. Since $\tau_f = \tau_s + \tau_a$, we need to show that (i) $\sum_{i=2}^{n} \tau_a^i \le (n-1)\tau_a + \gamma$.

To prove (i), we first prove by induction that (ii) $\alpha^n = \alpha^1 + \sum_{i=2}^{n} \tau_a^i - (n-1)\tau_a$ holds for every $n \ge 1$. For $n = 1$ (ii) obviously holds. Suppose (ii) holds for $n = n'$, namely that (iii) $\alpha^{n'} = \alpha^1 + \sum_{i=2}^{n'} \tau_a^i - (n'-1)\tau_a$. We will now show that it is correct for $n = n'+1$ as well; namely, that $(*)\alpha^{n'+1} = \alpha^1 + \sum_{i=2}^{n'+1} \tau_a^i - n'\tau_a$. To this end, we distinguish between the following two cases:

- **$0 \le \alpha^{n'} < \tau_a$:** In this case, by step 2 of the SCS algorithm follows that (iv) $\hat{\tau_a}^{n'+1} = \tau_a - \alpha^{n'}$. From steps 4.2 and 4.3 of the SCS algorithm it also follows that: (v) $\alpha^{n'+1} = \tau_a^{n'+1} - \hat{\tau_a}^{n'+1}$. From (iv) and (v) follows that (vi) $\alpha^{n'+1} = \tau_a^{n'+1} - \tau_a + \alpha^{n'}$, and by (vi) and the induction assumption (iii) the claim $(*)$ holds.

- **$\alpha^{n'} \ge \tau_a$:** In this case, by step 2 of the SCS algorithm follows that $\hat{\tau_a}^{n'+1} \le 0$. Therefore, by step 3 of the SCS algorithm (vii) $\tau_a^{n'+1} = 0$ and $(viii)$ $\alpha^{n'+1} = \alpha^{n'} - \tau_a$. Finally, from (vii), $(viii)$ and the induction assumption (iii) follows that $(*)$ holds for this case as well.

After having proven (ii), we complete the proof of the theorem by using (ii) in order to prove (i). Since from (ii) follows that $\sum_{i=2}^{n} \tau_a^i = \alpha^n - \alpha^1 + (n-1)\tau_a$, then in order to prove (i) we need to show that (ix) $\alpha^n - \alpha^1 \le \gamma$. However, since for every i $0 \le \alpha^i \le \gamma$ holds (this can be shown by a simple induction on i: for $i = 0$ $\alpha^i \overset{\Delta}{=} 0$, and by the SCS algorithm α^i can be set to 0 in 4.2.2, or set to $e < \gamma$ in 4.3.2, or set to a value which is $< \alpha^{i-1}$ but ≥ 0 in 3.2), then (ix) holds and the theorem is correct. $\square$

Theorem 3
A synchronous source that is assigned s slots in the synchronous region of every frame has a dedicated synchronous channel with a bandwidth of $T \cdot s/\tau_f$, where T is the transmission rate on the upstream channel.

Proof
Follows directly from Theorem 2. $\square$

[3]Note that in the SCS algorithm (Figure 7), FRAME-1 represents the first frame after the system was configured with the current values of τ_a and τ_s, and therefore α^0 is defined as 0. In this theorem, in order to simplify the notations, it is assumed that FRAME-1 is the first frame of the considered synchronous call, which is not necessarily the first frame since the system was configured with the current values of τ_a and τ_s. This implies that for the sake of this proof α^0 might be greater than 0.

7 Conclusions

The paper has addressed the problem of guaranteeing quality of service to synchronous sources on the upstream channel of an MXL CM network. The paper has shown that The MXL is a unique MAC protocol that does not require that packets will be broken down into a stream of fixed sized units at the sending CM and be re-assembled after delivery. Consequently, the support of synchronous traffic becomes a much difficult challenge. A naive solution, where the SCS divides the time domain into a series of successive frames, and every frame is divided into fixed-size synchronous and asynchronous regions, would result in loss of asynchronous bandwidth when the number of asynchronous slots remaining in some frame is less than the number needed for the transmission of an asynchronous packet by the reserving station. The paper presented a new scheme where the SCS is allowed to extend the asynchronous part of a frame. However, in order to keep the average size of a frame fixed, the SCS will have to reduce the size of the asynchronous part in subsequent frame(s) by the same number of slots. The paper has shown that the new scheme guarantees the synchronous stations the same quality of service provided by the FDDI timed token protocol. That is, a guaranteed average delay between consecutive transmissions, a guaranteed maximum delay between consecutive transmissions, an a guaranteed bandwidth on the upstream channel.

8 References

[1] A. Banerjea, D. Ferrari, B. Mah, M. Moran, D. Verma, and H. Zhang. The TENET real-time protocol suite: Design, implementation and experiences. *IEEE/ACM Transactions on Networking*, 4(1), February 1996.

[2] C. Bisdikian, B. McNeil, R. Norman, and R. Zeisz. MLAP: A MAC level access protocol for the HFC 802.14 network. *IEEE Communications Magazine*, 34(3), March 1996.

[3] CCITT. *Specifications of Signalling System No. 7, Recommendations Q.700-Q.716*, November 1989.

[4] J. Dail, M. Dajer, C. Li, P. Magill, C. Siller, K. Sriram, and N. Whitaker. Adaptive digital access protocol: A MAC protocol for multiservice broadband access networks. *IEEE Communications Magazine*, 34(3), March 1996.

[5] R. Gusella. A measurement study of diskless workstation traffic on an ethernet. *IEEE Journal on Selected Areas in Communications*, 38(9), 1990.

[6] M. Johnson. Proof that timing requirements of the FDDI token ring protocol are satisfied. *IEEE Transactions on Communications*, 35(6):620–625, June 1987.

[7] R. Chiu, R. Cohen, Michael Chen and B. Hutchinson. The MXL MAC protocol for HFC networks. HP-Labs TR.

[8] D. Sala and J. Limb. A protocol for efficient transfer of data over a fiber/cable systems. In *INFOCOM*, March 1996.