RC 20284 (12/12/95)
Computer Science/Mathematics 23 pages

IBM Research Report

MLAP: A MAC Level Access Protocol for the HFC 802.14 Network

Chatschik Bisdikian

IBM Research Division
T.J. Watson Research Center
Yorktown Heights, New York

Bill McNeil, Rob Norman, and Ray Zeisz

IBM Corporation
Networking Hardware Division
3039 Cornwallis Road
Research Triangle Park, NC 27709

IBM Research Division
Almaden · T.J. Watson · Tokyo · Zurich

MLAP: A MAC Level Access Protocol for the HFC 802.14 Network

Chatschik Bisdikian

IBM Research Division
T. J. Watson Research Center
30 Saw Mill River Road
Hawthorne, NY 10532, USA

Bill McNeil, Rob Norman and Ray Zeisz

IBM Corporation
Networking Hardware Division
3039 Cornwallis Road
RTP, NC, 27709, USA

Abstract: *This paper describes the MLAP MAC protocol for sharing the bandwidth of an HFC network. MLAP is a flexible ATM friendly MAC protocol, capable of supporting various types of traffic with diverse quality-of-service requirements. MLAP MAC functions such as: message formats, bandwidth requests/grants mechanisms, collision resolution, synchronization, support for varying PHY layer options (modulation rates, types, preambles) will be presented. Finally, new station power-on procedures will also be discussed.*

1 Introduction

Interactive residential broadband/multimedia services are expected to be the next main event in the cyberspace experience. Services like video-on-demand (VoD) (where people will choose movies of their liking to watch at any time of their choice), effective tele-commuting (by bringing the full power of one's familiar office computer interface to one's home), remote learning, tele-shopping, the ever increasing on-line newsgroup interactions, Web surfing, remote video-game playing, etc, are waiting to jump into people's homes as soon as the 28.8kbps barrier of today's fastest modems is toppled 10-fold, 100-fold, or even more, by a new breed of communication technologies. The great interest in providing effective on- and off-ramps from people's homes to "information superhighways" is only underlined by the plethora of publications in trade and technical journals, [4, 5, 6, 7, 8] and reference therein.

The large excess bandwidth (well over 300 MHz) available in today's cable TV (CaTV) hybrid fiber/coaxial (HFC) plants is an ideal candidate to provide the underlying communications infrastructure for interactive digital services to the home; cable operators that have not yet upgraded their all-coaxial plants to HFC are quickly moving in this direction.

In order to maintain compatibility with today's analog TV channels, the spectrum (of a CaTV plant) between 54 MHz to 450 MHz is not going to be affected by the new services. The spectrum between 5 MHz to 42 MHz will be used for "upstream" transmissions, i.e., from the home to the CaTV *headend* station and from there to the rest of the world, while the spectrum between 450 MHz to 750 MHz will be used for "downstream" transmissions. Since the theoretical frequency bandwidth of the cable is in the order of a GHz, future generations of this technology may have a different split and frequency ranges between upstream and downstream transmissions.

To cope better with power limitations, cost, channel error characteristics, system management, etc, the downstream and upstream spectra will be further subdivided into a number of channels, for example, 1-2 MHz wide for each upstream channel, and 6 MHz wide for each downstream channel. Each of these channels, using appropriate modulation techniques, will be capable of carrying possibly from 2-10 Mbps (or more) of digital data on the upstream (shared by up to a few thousand stations), and 30-40 Mbps (or more) on the downstream (a broadcast channel).

Due to the "peculiarities" of CaTV tree and branch topology (e.g., the splitting of the receive and transmit frequencies), the various stations cannot directly listen to the upstream transmissions from other stations and hence they are incapable of detecting collisions and ultimately coordinating their transmissions all by themselves. The headend station is the focal point of the tree and branch topology and plays a major role in coordinating the access to the upstream channel. In addition to coordinating upstream transmissions, since the downstream channels are broadcast (one-to-many), the headend station also has complete control of the downstream transmissions, as well.

The IEEE802.14 working group (WG) has been formed with the responsibility of standardizing physical layer (PHY) and medium access layer (MAC) protocols for HFC networks. The MLAP is a MAC level protocol for the HFC environment, that has been formally submitted to the IEEE802.14 WG for consideration. MLAP operates on a full-duplex, multichannel, shared media network consisting of a variable number of stations spread over several kilometers and a single headend controller located several tens of kilometers away from the stations. MLAP is a flexible "ATM-friendly" protocol that supports *quality-of-service* (QoS) constraints for various classes of traffic. It allows for contention-based and reservation-based transmissions which are dynamically adjusted to the traffic conditions and has provisions to operate over several modulation techniques. The MLAP is supported by a management protocol that provides (among other things) for a ranging and joining process in order to add dynamically newly powered-on stations in a "plug-and-play" manner.

In the following sections of this paper, we will present the key features of MLAP. We start first with the HFC network model.

2 The HFC network model

The CaTV network topology is summarized in figure 1. The topology is a collection of tree and branches with the root of the tree located at the CaTV headend station, hereafter to be called simply the headend, and the leaves at people's homes. In an HFC network, we identify two domains: (a) the *fiber domain*, that extends from the headend to the customer's neighborhood containing anywhere from 500 to 2000 homes, and (b) the *coax domain*, which covers the "last-mile(s)" distribution network to the customers. The distance from the headend to the farthest home is about 80 km, with the homes located only at about the last 20% (or less) of this distance.

Historically, the whole CaTV (distribution) network was comprised just of coaxial cables. The advantages of the fiber cable, such as reduced signal attenuation, smaller number of signal amplifiers required, resistance to electromagnetic interferences, and larger bandwidth, have "encouraged" many cable operators to upgrade their plants from just coaxial cabling to the hybrid fiber/coax cabling. The *fiber nodes* at the edges of the fiber and coax domains are responsible for converting optical signals to electrical for the downstream transmissions, and vice versa for the upstream transmission. Also, since traditionally, CaTV transmissions have been broadcasted from the headend to *all* the customers, the amplifiers on the network were

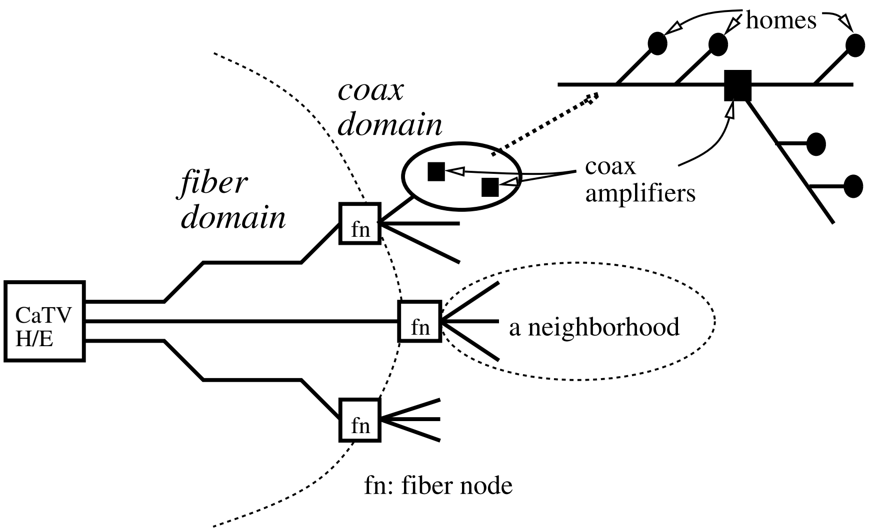

Figure 1: The CaTV network topology

one-directional as well. Bi-directional transmissions on the HFC network will require the upgrade of the one-directional signal amplifiers to bi-directional. This is not as burdensome as it sounds, since an HFC network uses a substantially lower number of signal amplifiers (relative to a just coaxial plant), and since cable operators can add the bi-directional amplifiers at the same time that they upgrade their plants to HFC.

3 The MLAP protocol

The tree and branch CaTV topology, that has grown over several decades of cable TV deployment, is ideally suited for broadcast transmissions. This topology possess a challenge in developing efficient MAC protocols, especially for the many-to-one transmission on the upstream channels. Stations cannot listen to each others transmission directly and hence they cannot self-regulate their access to the network. The headend station on the other hand can play a major role in coordinating the station transmissions. The MLAP protocol was developed to take advantage of the "peculiarities" of the HFC network in an efficient, flexible, and robust manner.

The basic philosophy behind MLAP is summarized in figure 2. Inactive stations, i.e., stations with nothing to transmit, do not need to do anything. When new traffic is generated in a station, then the station attempts to notify the headend of its new traffic through the use of a random access mechanism. For this reason, the headend periodically notifies all the stations, through transmissions on the downstream channel, of time instances and

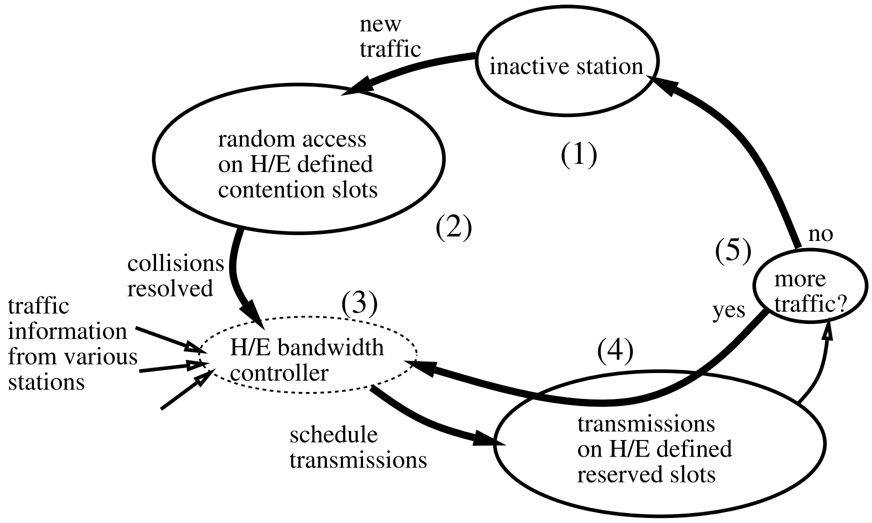

Figure 2: The MLAP philosophy

time lengths that contention-based transmissions can occur on an upstream channel. When collisions are successfully resolved and traffic information reaches the headend, a bandwidth controller at the headend schedules future transmissions from the stations. In order for the network to support an abundance of services, the traffic scheduler at the headend will be guided by QoS constraints. Once again, stations are notified on the downstream of time instances and time intervals that prescheduled reservation-based transmissions can occur. Contention-based and reservation-based transmissions do not interfere with each other. If an active station generates additional traffic during the period of its reservation-based transmissions, the station can use the (collision-free) reservation-based transmissions to convey to the headend information regarding the additional traffic.

Note that the specific traffic scheduling algorithm used by the bandwidth controller is not part of MLAP. We need only to assume that the scheduling algorithm can prioritize transmissions based on service class information sent by stations. Of course, the headend must also comply with the MLAP primitives to be discussed in the next section.

Before proceeding with the detailed presentation of MLAP, we will first summarize some of the key features of MLAP:

- MLAP supports integrated services: in particular, MLAP
  - encapsulates ATM cells in MLAP frames, i.e., MLAP is ATM "friendly;"
  - supports multiple priorities of traffic to be used appropriately by various classes of traffic (e.g., ABR, CBR, UBR, VBR) to enable QoS guarantees;

- MLAP achieves efficient use of the upstream channel by supporting:
  - flexible contention and reservation modes of operation, where newly activated stations content to "establish themselves," and then transmit on a reservation until they "empty their queues;"
  - headend control of contention-/reservation-based transmissions based on traffic priority considerations, i.e., headend dictates who transmits where and when;
  - variable length frames (without compromising QoS constraints);
  - minimize bandwidth waste due to collisions through the use of contention minislots;
  - flow control of traffic entering the system;
- MLAP supports multiple PHY options by allowing several modulation rates, types, preambles, etc;
- for each downstream channel, MLAP supports several upstream channels.

4 The MLAP protocol primitives

The MLAP protocol is a *blocked* protocol in that the downstream and upstream channels are delineated by a constant time intervals, hereafter called the *block*.[1] The block duration is required to be longer than the worst case round-trip delay of the network including all delays (propagation, headend, and station processing times). The block duration must be established by the network operator as a headend-controller parameter, and it is typically a few milliseconds long. Blocks are further divided into variable size *slots*. Each slot contains a message, hereafter to be referred to as a *primitive*.

It is assumed that during initialization, each station goes through a *ranging* process to adjust its upstream block in such a way, that the upstream blocks from all stations are exactly aligned when they reach the headend; for more detail see Appendix A.

Detailed description of the MLAP primitives will be given shortly. We first give a short description of each one of them in the order "experienced" by a station ready to transmit.

DS.SYNC Enables the station to maintain synchronization with the start of the downstream block.

[1] In the literature such systems are commonly referred to as *framed* systems. Since the word "frame" also refers to a network layer data unit, we prefer to use the word "block" to denote a framed protocol.

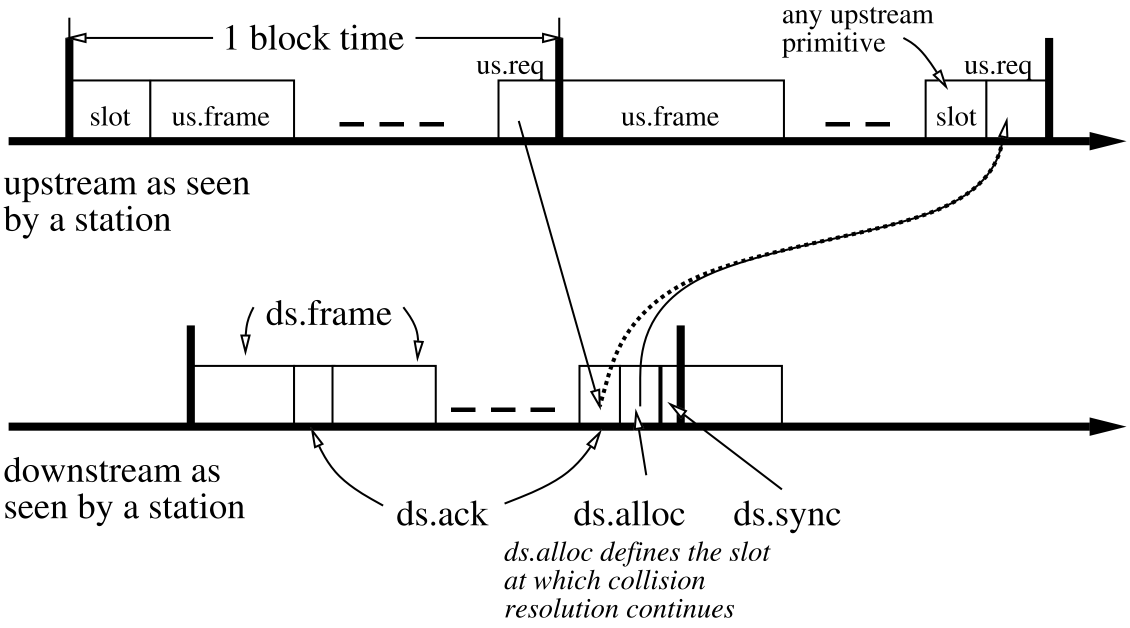

Figure 3: MLAP and contention-based transmissions

DS.ALLOC Carries upstream slot allocations, i.e., defines contention and reservation slots. The entire upstream block format is determined solely by DS.ALLOCs.

US.REQ Carries a bandwidth request.

DS.ACK Carries the results of contention-mode transmission attempts.

US.FRAME Carries payload(s) (data, or MAC management control information) and request for additional upstream transmission bandwidth (RB).

DS.FRAME Carries a payload (data, or MAC management control information).

All primitives are aligned on 8-bit boundaries for ease of implementation. Figures 3 and 4 depicts examples of the use of the primitives. Figure 3 shows how contention-based transmissions using US.REQs are coordinated with the help of DS.ACK and DS.ALLOC primitives. Figure 4, on the other hand, shows how a *chain* of collision free reservation-based transmissions can extend over several blocks with the use of the *request bandwidth* (RB) field in a US.FRAME slot

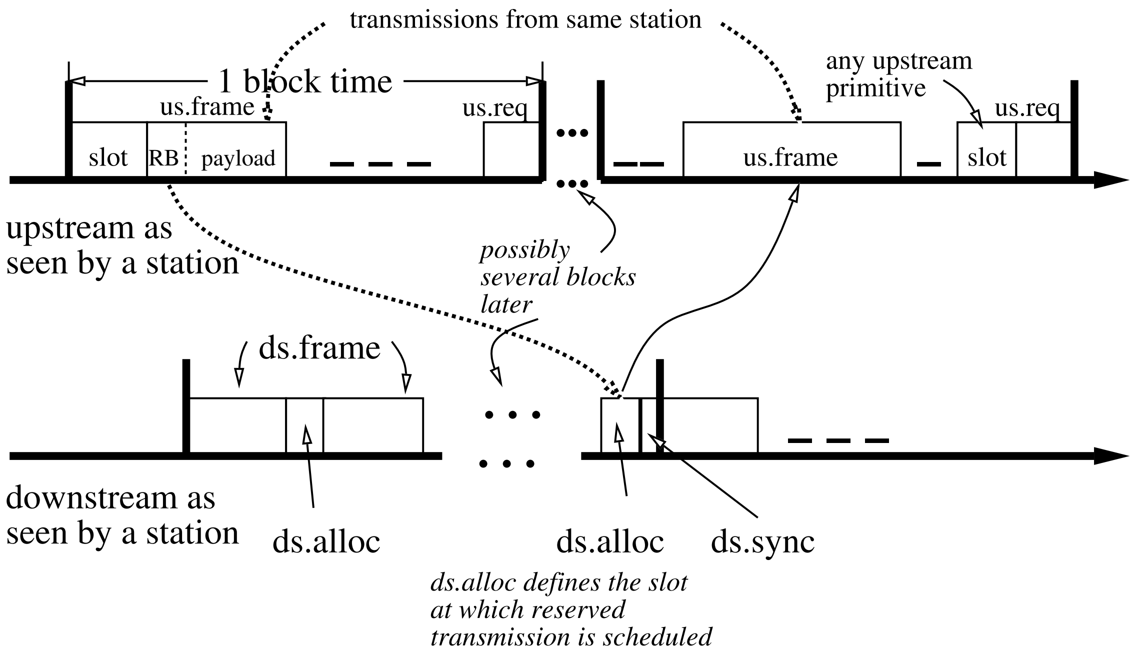

Figure 4: MLAP and reservation-based "chain" transmissions

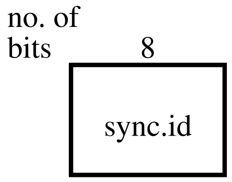

Figure 5: The DS.SYNC primitive

4.1 The downstream primitives

The downstream primitives are sent from the headend to the stations on a downstream channel and contain actual (application) data, or MAC control information.

4.1.1 The synchronization primitive DS.SYNC

The DS.SYNC primitive is sent in the downstream to provide a synchronization marker to allow the station to align its downstream block boundaries with the headend controller. The DS.SYNC primitive consists of the 8-bit pattern 'xD8',[2] that serves as the DS.SYNC id, see figure 5.

[2]The notation 'xnn' implies the hexadecimal number nn.

no. of bits	8	16	10	2	8	8	12	16
	alloc.id	source address	offset	prob–ability	fre–quency	PHY control	grant	CRC

Figure 6: The DS.ALLOC primitive

4.1.2 The allocation primitive DS.ALLOC

The DS.ALLOC primitive is sent in the downstream to define a slot in the *next* upstream block. Each DS.ALLOC primitive defines one slot in the next upstream block and each slot can carry one upstream primitive. The DS.ALLOC contains all the information necessary to control where in the upstream block the slot is located, which stations may access the slot, as well as the mode, i.e., contention or reservation, in which they may access it. Each DS.ALLOC primitive begins with the pattern 'xD9' and is protected by a 16 bit CRC field. The following information is contained in the primitive, see figure 6.

- **source address:** The *source address* field specifies the station or group of stations that may access the specified slot. If the slot is reserved for a particular station, the field will contain the individual address of that station. If the slot is a contention slot, the field will either contain a group address or the broadcast address. The source address is a "short" MLAP address assigned to a station during power-on, see Appendix A.

- **offset:** The *offset* field specifies, in units of 1/1024-th of the upstream block length, the starting point for the slot. A value of zero indicates that slot begins at the start of the upstream block.

- **probability:** The *probability* field specifies the random probability, see table 1, with which all newly-contending stations will transmit in the contention slot defined in the *grant* filed, see below. The *probability* field allows MLAP to flow control newly arriving

probability field:	00	01	10	11
"admission" probability:	0%	25%	50%	100%

Table 1: The *probability* field interpretation

traffic to the system.

- **frequency:** The *frequency* field specifies the frequency to be used for transmitting in the slot. For a 5-42 MHz upstream spectrum the *frequency* field specifies upstream frequencies at 150 KHz steps. The *frequency* field allows MLAP to support multiple upstream transmission channels per single downstream channel.

- **PHY control:** The *PHY control* field specifies the modulation rate, modulation type, and preamble type to be used for transmitting in the slot. Two three-bit subfields are used to specify the modulation rate and the modulation type. The remaining 2 bits specify the preamble type to be used in the upstream slot. The *PHY control* field allows MLAP to support a multitude of PHY layer implementations.

- **grant:** The *grant* field specifies the number of payload(s) that may be sent in an upstream slot and the service class. The allocation count subfield is 4 bits and specifies 0 to 15, 53-byte allocation(s). A value of zero indicates that the upstream slot may contain only the bandwidth request primitive, i.e., a US.REQ. The *service class/contention-id* subfield is 8 bits. If the allocated slot is a reservation slot, then this subfield specifies the service class for which the grant is given. If the slot is a contention slot, then this field contains a *contention-id* number that is correlated with the DS.ACK primitives that are sent to indicate collision status of contention slots. The *grant* and *offset* fields combination allows MLAP to define variable size transmission slots in a block for optimal upstream bandwidth usage without compromising QoS.

- **CRC:** The *cyclic redundancy check* sequence is generated by the CRC-CCITT polynomial $x^{16} + x^{12} + x^5 + 1$, and covers the entire primitive including the first byte.

4.1.3 The acknowledgement primitive DS.ACK

The DS.ACK primitive is sent in the downstream to indicate the status of a contention slot. The DS.ACK contains the information necessary for contending stations to determine whether a collision has occurred in a particular slot. DS.ACK primitives are not sent for reserved slots because no collisions are allowed in these slots. For fast response, DS.ACK primitives are sent asynchronously with respect to block timing, see figure 3, however each primitive contains a contention id number that may be used by the station to relate the acknowledgment to a particular upstream contention slot. Each DS.ACK primitive begins with the pattern 'xDA' and is protected by a 16 bit CRC field. The following information is contained in the primitive, see figure 7.

no. of bits	8	16	8	8	8	16
	ack.id	source address	fre–quency	cont.id	status	CRC

Figure 7: The DS.ACK primitive

- **source address:** The *source address* field specifies the group of stations that the acknowledgment is for. The field will either contain a group address or the broadcast (short MLAP) address.

- **frequency:** The *frequency* field specifies the frequency of the channel carrying the contention slot. Same use as in a DS.ALLOC.

- **cont.id:** The *contention-id* field specifies which contention slot is being acknowledged. The contention-id number is assigned by the headend controller in the grant field of a DS.ALLOC primitive and is unique per upstream channel. The *contention-id* field allows MLAP to support, whenever needed, a dynamically changing number of contention slots per block.

- **status:** The (slot) *status* field contains a two-bit subfield that specifies the status of a contention slot, according to table 2. The remaining 6 bits are reserved. The

status field:	00	01	10	11
slot status:	empty	no collision	reserved	collision

Table 2: The *status* field interpretation

START-n collision resolution algorithm used in MLAP, see section 5, requires only binary "Collision/No_collision" (C/NC) feedback. The other values shown above provide compatibility with other algorithms.

- **CRC:** Same use as in a DS.ALLOC.

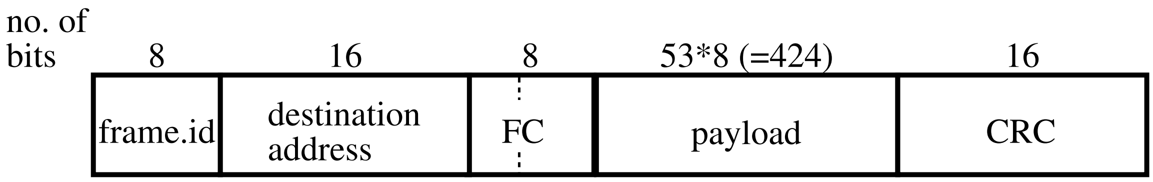

Figure 8: The DS.FRAME primitive

4.1.4 The frame primitive DS.FRAME

The DS.FRAME primitive defines an MLAP downstream frame. All downstream frames are identical in size, however, they carry two different payload types: management payloads and data (ATM cell) payloads. Each DS.FRAME primitive begins with the pattern 'xDB' and is protected by a 16 bit CRC field. The following information is contained in the primitive, see figure 8.

- **destination address:** The *destination address* field specifies the station or group of stations that the frame is sent to. The field will either contain an individual, group, or broadcast (short MLAP) address.

- **FC:** The *F*(*rame*) *C*(*ontrol*) field specifies information about the payload of the frame. The first bit of the *FC* field indicates the payload type as follows: 0 - data (ATM cell), 1 - management. The second bit of the FC field specifies encryption of the payload as follows: 0 - no encryption, 1 - encrypted. The remaining bits are reserved.

- **payload:** The *payload* field contains one 53-byte payload that is either a data (ATM cell) payload or a MAC management payload.

- **CRC:** Same use as in a DS.ALLOC.

4.2 The upstream primitives

The upstream primitives are sent from the stations to the headend on an upstream channel and contain actual (application) data, or MAC control (management) information.

4.2.1 The frame primitive US.FRAME

The US.FRAME primitive defines an MLAP upstream frame. Upstream frames carry two different payload types: management payloads and data (ATM cell) payloads. US.FRAME may be allocated (and therefore sent) in either a contention slot or a reservation slot. US.FRAMEs

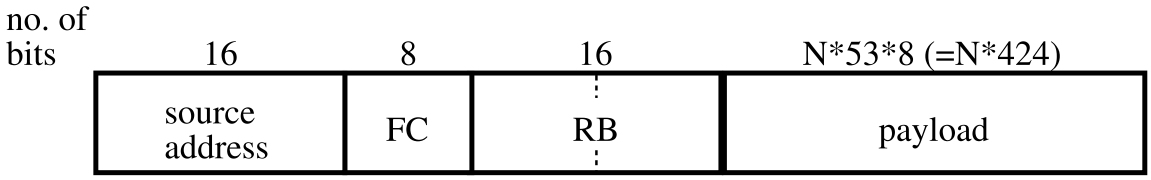

Figure 9: The US.FRAME primitive

that are allocated as the result of a station request will always occur in a reservation slot. The headend may allocate US.FRAMEs in contention slots in order to allow payload(s) to accompany a station request.

This primitive is assumed to be encapsulated in an upstream PHY frame, with error detection and correction (if required) being the responsibility of the PHY. The following information is contained in the primitive, see figure 9.

- **source address:** The *source address* field specifies the station that the frame is sent from. The field will always contain an individual (short MLAP) address.
- **FC:** Same use as in a DS.FRAME.
- **RB:** The *R(equest) B(andwidth)* field provides the headend with the information (queue-size and traffic class) needed to describe a request for bandwidth from the station. The queue depth subfield is 8 bits and specifies 0 to 255, 53-byte values. A value of zero indicates that no bandwidth is being requested. The service class sub-field is 8 bits and specifies the service class for which the request is made.
- **payload:** The *payload field* contains either data (ATM cell) payload(s) or MAC management payload(s). The number of payload(s) is defined by the *grant* field of the corresponding DS.ALLOC primitive sent in the downstream. The number of 53-byte payload(s) may range from 1 to 15. A value of zero in the *grant* field indicates a US.REQ (instead of US.FRAME) primitive in the slot.

4.2.2 The request primitive US.REQ

The US.REQ primitive is sent in the upstream to indicate a request for reservation mode slot(s). US.REQs are always allocated (and therefore sent) in contention slots. This primitive is assumed to be encapsulated in an upstream PHY frame, with error detection and correction (if required) being the responsibility of the PHY. The following information is contained in the primitive, see figure 10.

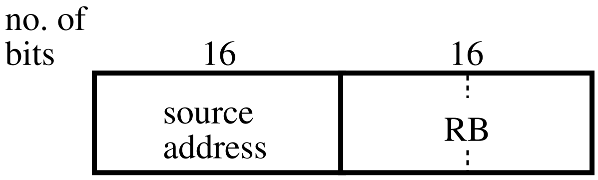

Figure 10: The US.REQ primitive

- **source address:** Same use as in a US.REQ.
- **RB:** Same use as in a US.REQ.

Since US.REQs are much smaller than US.FRAMEs, they minimize bandwidth waste due to collisions. A successful transmission of a US.REQ could initiate a *chain* of collision-free reservation-based US.FRAME transmissions.

5 Random access and collision resolution

5.1 The START-n algorithm

One of the most challenging tasks for a station is to convey successfully to the headend information about its traffic reserves. After an initial "contact" with the headend, the station could continue to transmit in a chain of collision-free transmission under the guidance of the headend, see figure 4.

A station that has remained inactive, i.e., it did not have any traffic to transmit, for sufficient long before it becomes active again, it needs to "re-initiate" its chain by randomly accessing the upstream channel. For this, the station needs to observe DS.ALLOCs to determine when and where US.FRAMEs or US.REQs are available on a (random access) contention basis.

Since, collisions could occur on contention-based transmissions, a random access protocol is needed to coordinate the transmissions of newly activated stations. MLAP uses the *n-ary STAck ResoluTion* (START-n) algorithm as its collision resolution algorithm. START-n is a generalization of the n-ary stack algorithm [11, 3, 13] allowing it to operate in an MLAP environment where the contention-based opportunities may change from block to block.

The START-n algorithm requires that stations involved in a collision (contending stations) resolve the collision in subsequent upstream blocks using the slot corresponding to the same contention-id where the collision originally occurred. In the sequel, when we refer to the "next," or "subsequent," etc, slot, we will imply the slot in the next block with the same contention-id.

According to the START-n algorithm, a station, with new traffic to transmit, utilizes a (START) counter which the station updates according to feedbacks received regarding transmissions in successive slots. The counter is initialized at the value 0, and whenever the counter has this value, the station is permitted to transmit in the next slot.

Suppose that a station has already made its first transmission, at the end of slot it performs the following algorithm:[3]

```
if ( collision ) then
    if ( station transmitted ) then
            counter = random[0, ..., n − 1];
        else  counter = counter + (n − 1) ;
  elseif ( no_collision )
    if ( station transmitted ) then
            done;
        else  counter = counter − 1;
endif
if ( counter = 0 ) transmit in the next slot;
```

The operation of the START-n algorithm is illustrated in figures 11 and 12 for $n = 3$. For the illustration, we use a logical stack as a visual aid. The stack is *not* part of the algorithm. Referring to the figures, each position in the stack represents packets from all stations that share the same "wait time" until transmission, i.e., the same START-counter value. For instance, at time "a" at the left side in figure 11, the stack contains some number of packets (Z) from multiple stations that are next in line to be transmitted (stack position 0). Those packets labeled Y are next in line to be transmitted, and are located in position 1. Those packer labeled W are next, etc.

Assume Z is greater than 1, so when the stations sent the "Z" packets, a collision occurs. Each station that attempted transmission picks a random number from 0 to $2\,(= n-1)$ and places its packets in the corresponding stack position, as shown at time "b" at the right side in figure 11. The number of stations that were in position 0 at time "a" and have chosen position 0 to move to at time "b" are signified by "Za" in the figure, and so on. Each station that did not transmit adds $2\,(= n-1)$ to its START-counter and places the packets in the resulting stack position. The other possible scenario is illustrated in the bottom of the next figure. At time "c" at the left side in figure 12, there is only one station's packet

[3]The operator $random[\cdot]$ performs the random selection of its arguments.

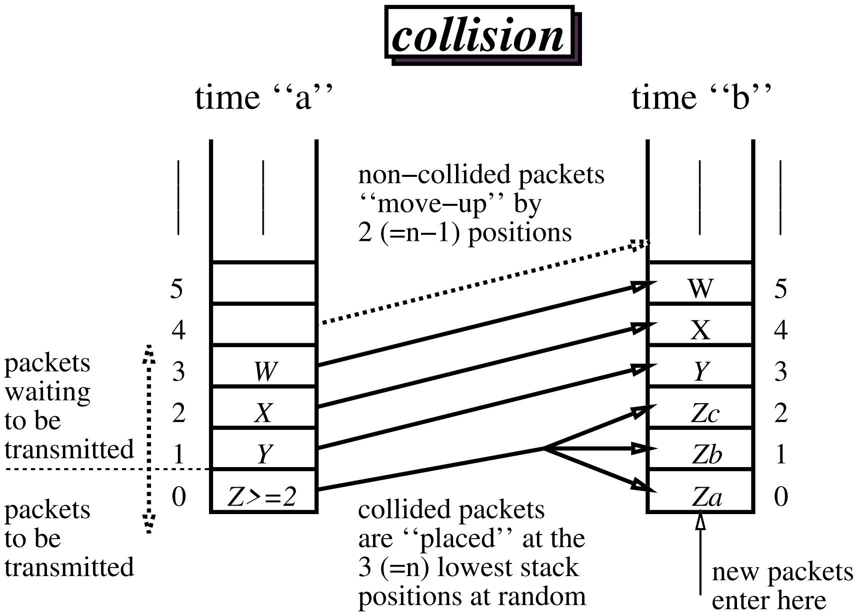

Figure 11: The START-n logical stack before and after a collision slot ($n = 3$)

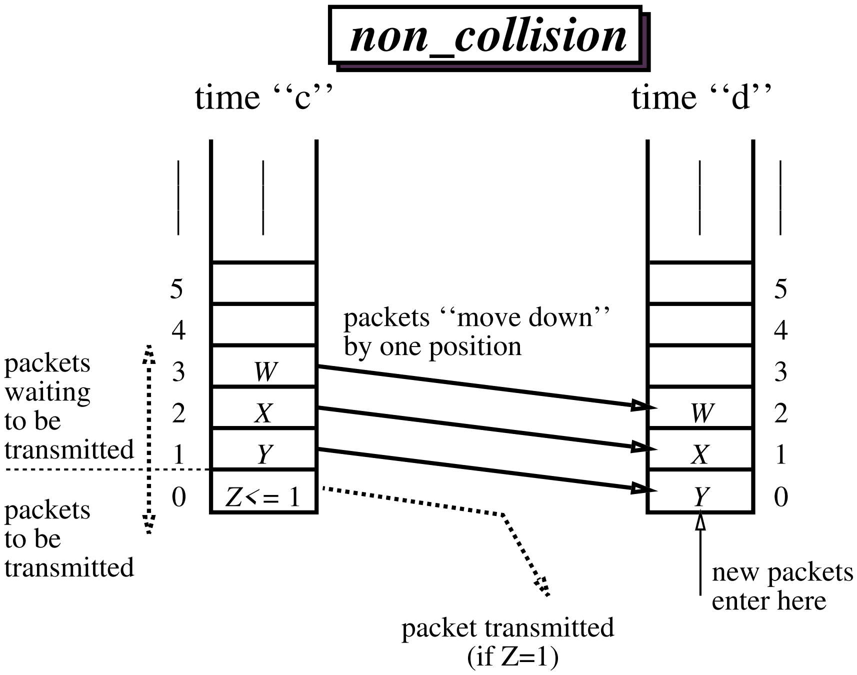

Figure 12: The START-n logical stack before and after a non-collision slot ($n = 3$)

n	2	3	4	5
arrival rate (cells/slot)	< 0.3601	< 0.4015	< 0.3992	< 0.3872

Table 3: START-n throughput for various n.

in position 0 ($Z = 1$), so the transmission is successful. This station that had a successful transmission now exits the collision resolution process. All other stations subtract 1 from their START-counters and place their packets in the resulting stack position, as shown at time "d" at the right side in figure 12. The process repeats until the stack is empty. New packets enter at position 0, providing a non-blocking entry to newcomers.

The stability conditions and the delay characteristics of START-n for fixed size slots and single-cell Poisson arrivals have been studied in [3, 11, 13]. Borrowing from their results, for Poisson arrivals, START-n is stable, i.e., all packets get to be transmitted successfully, whenever the arrival rate of the Poisson arrivals is as shown in table 3.

The maximum stable throughput is achieved for $n = 3$, and hence we assume START-3 as the default choice for implementing START-n. The high stable throughput of START-3 is higher than the (stabilized) ALOHA:[4] 0.368. It is also higher than the blocked-access tree search algorithms with the obvious first transmission rule, i.e., newcomers wait until the current collision resolution interval ends, and then all of them attempt to transmit. These algorithms provide 0.347 for binary feedback, and 0.375 for ternary feedback [10]. Note that START-n requires only binary feedback, and it does not block newcomers. A ternary feedback version of START-n has a stable throughput of 0.4076 [11], but it will not be considered further because the added complexity does not warrant the marginal throughput increase.

5.2 The START-n algorithm implementation

In MLAP, a station is allowed to operate several independent "START-engines" simultaneously with the following restrictions. The station must maintain a unique START-counter value for each concurrent contention resolution that the station will participate in. The number of concurrent contention resolutions is limited by the number of service classes implemented at the station. Moreover, there can be no more concurrent contention resolutions

[4]ALOHA protocols based only on acknowledgements have a theoretical throughput of 0 [1, 2, 9, 15]! Stabilized versions of "ALOHA" [2, 15] require at least a binary C/NC feedback information, which is similar to the feedback in START-n.

than there are slots in an upstream block. Each START-engine is associated with one and only one contention-id, and also each service class is allowed to use at most one contention-id.

Prior to participating in a contention resolution, a station has to select (if allowed) a contention slot, or, in other words, select a contention-id. There are various schemes for achieving this. The simplest one is for a station to select randomly a contention-id and then use the *probability* (flow control) field associate with this contention-id to see whether it is allowed to use it. If it is not allowed, the station will repeat the contention-id selection process again in the next block; other contention-id selection schemes are also possible. Note that a station is not allowed to reuse a contention-id already in use by the station for an on-going collision resolution. Finally, a contention-id is released by the station, when its collision on the corresponding contention slot has been successfully resolved.

5.3 Removing/Adding contention slots

During normal operation, the headend controller should guarantee the existence of a contention slot with the same contention-id in future blocks until the contention is resolved. When a collision resolution in a contention slot is complete, i.e., there are no stations that are using the particular contention slot to resolve their collisions in the slot, the headend may decide to remove this contention slot (and its contention-id), or leave it to be used again by newly activated stations.

To determine when a collision resolution in a contention slot has terminated, the headend controller will maintain a counter for each contention-id in use. The initial value for these counters is 1. Following the rules of START-n, it can be easily shown [13] that the first time that a headend counter reaches 0, the collision resolution in the corresponding contention slot has terminated.

In exception situations, e.g., when extra upstream bandwidth needs to be dedicated urgently to reservation-based transmission, it is allowable for the headend controller to remove a contention-id (and corresponding slot) during an on-going contention resolution. The contending stations will detect this by the absence of the contention-id from the set of DS.ALLOC(s) sent during a block. In this case, the contending stations will terminate contention resolution and start the contention transmission process over again as a newcomer, by selecting an available contention-id; other *remove* schemes are also possible. On the other hand, new contention-ids, and thus contention slots, may become available in a block, e.g., to accommodate an increase number of newcomers. The headend using the *probability* field may direct all, or some, of the newly activated stations to use these new contention slots, thus reducing the burden on the “old” contention slots; other *add* schemes are also possible.

6 The bandwidth request and allocation protocols

6.1 The bandwidth request protocol

The station implements an algorithm for bandwidth requests that prioritizes requests based on service classes. Each cell-queue that is implemented by the station is associated with a particular service class. The order of requests sent will be the order of priority of the service classes from 0 (highest) to 255 (lowest). The station MAC should limit requests to a maximum of one per service class per upstream block or the number of contention slots in the upstream block, whichever is less.

A request for a particular service class should indicate the queue depth at the time the request is made after being adjusted for any grants that were received (in DS.ALLOCs in the previous downstream block). If a reservation-mode US.FRAME of that particular service class has been allocated, the request is placed in the US.FRAME's *RB* field. If a reservation-mode US.FRAME's *RB* field is not being used by the service class matching its payload, then requests from other service classes (in priority order) may use the *RB* field. If a request is unable to use the *RB* field of a reservation-mode US.FRAME, then it will use the *RB* field of either a US.REQ (always contention-mode), or a contention-mode US.FRAME. Since a contention-mode US.FRAME always has at least one payload region, the station may also send payload(s) matching the service class, in which case the queue depth sent will be adjusted by the number of payloads sent.

The station chooses between US.REQ and contention-mode US.FRAME by using a "service class to request-type" mapping that is maintained at the MAC management level. This mapping will define if a service class is allowed to use a contention-mode US.FRAME. If the station is allowed, it will use contention-mode US.FRAME if available, otherwise it will use US.REQ.

6.2 The bandwidth allocation protocol

Downstream: In the downstream direction, ATM cells that are received across the MAC service interface are transmitted in available slots after being encapsulated in MLAP frames. There is no multiple access contention in the downstream direction.

Upstream: Upstream channels are controlled by a time division multiple access scheme based on slots. There is a mixture of slots that may be accessed on a contention basis and slots that are reserved for a particular station or group of stations at a particular service class. As part of the request mechanism, a station may request a recurring slot allocation by sending a special MAC management *stream-request* frame.

After allocating any permanent slots, upstream slots are allocated fairly to station MACs within each service class. Service classes are allocated slots based on priority, in order from 0 (highest) to 255 (lowest). Each service class is fully satisfied, in order, before any slots are allocated to lower service classes.

7 Concluding remarks

In this paper, we have introduced a very versatile MAC protocol for the HFC 802.14 network. MLAP can provide integrated broadband services to the home, internetwork easily with ATM wide area networks, support QoS constrains for various types of traffic, and operate over a variety of PHY layer protocols. Our presentation is only an extended summary of our medium sharing protocol proposal to the 802.14 WG. To keep our presentation simple, we have considered only error-free system operation. MLAP contains provisions for what happens when, say, DS.ACKs, or DS.ALLOCs are lost, it provides for the use of timers for the recovery of error situations in the network. MLAP is also supported by a management suit of protocols for the management of the overall system. In the appendix, we briefly present the procedures for managing the attachment of a new station to the network. The complete (as of this moment) suit of our MAC related protocols is provided in [12, 14, 16].

Although ATM has been pervasive in our presentation, the MLAP philosophy is not tied to ATM. ATM has been selected merely for its very advantageous service integration properties. Whole ATM cells are encapsulated in MLAP frames, and MLAP transports them transparently to the contents of the ATM cells. MLAP can be used over any system for which a basic unit of transmission has been established. This basic unit can be as small or as large as physical and other constraints permit, from a single bit to tens of ....

We are currently developing MLAP evaluation tools. Some early results are shown in figure 13, with more to come in future publications, when the 802.14 WG converges to mutually agreed upon network assumptions, traffic models, and performance metrics. The results in figure 13 concern a system with 16 stations, sharing a 2 Mbps upstream channel which consists of 2 msec long blocks. Each station generates two types of traffic: (a) a constant-bit rate (CBR) type of traffic, designated as a high priority traffic; and (b) an available-bit rate (ABR) type of traffic, designated as low priority traffic. 60% of the total amount of traffic generated by a station is assumed to be CBR traffic, while the remaining 40% is ABR. The figure shows the mean access delay, i.e., the time elapsed from the moment a message is generated in a station until it is placed in its entirety on the network. Note that the high priority CBR traffic maintains a desirable low latency even when the total offered loads becomes large. Although, in the region above 100% of total offered load the ABR

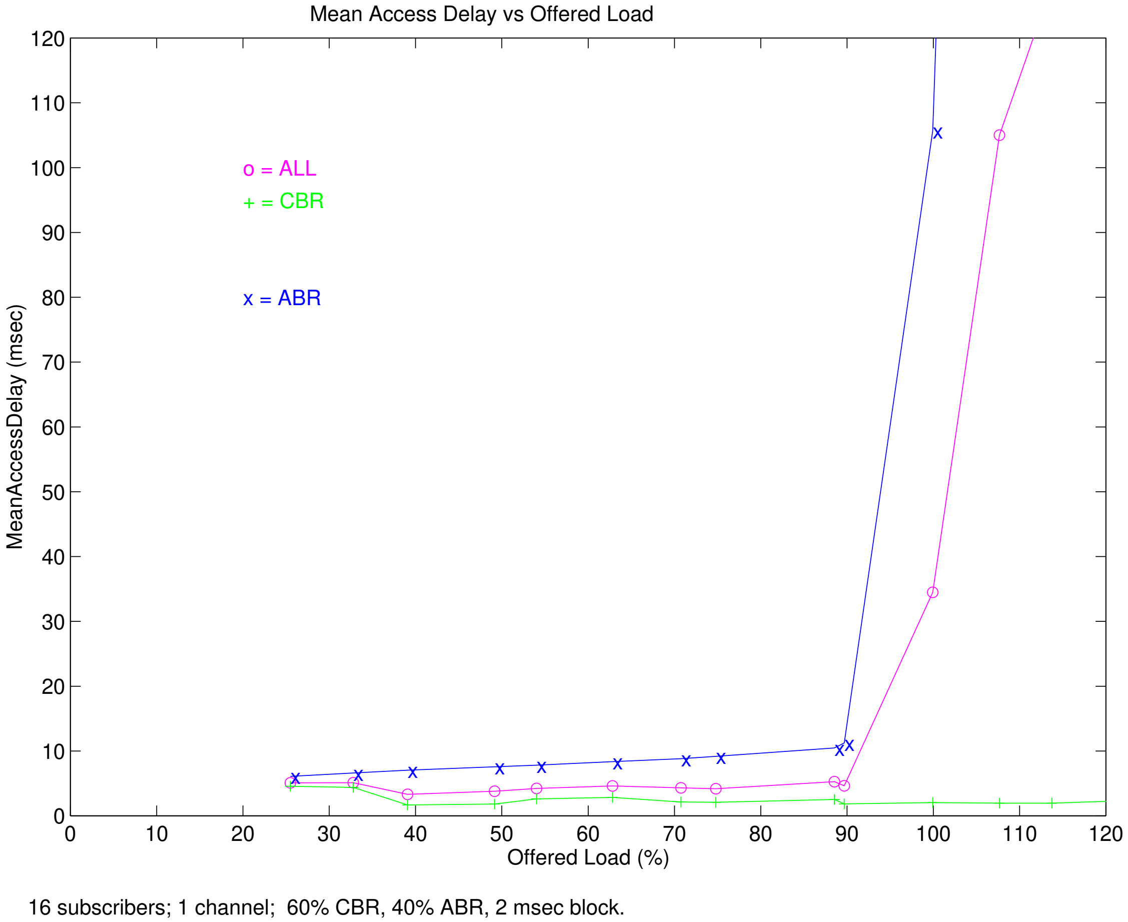

Figure 13: The mean access delay for in a 16 station system

traffic is penalized harshly, CBR traffic, whose offered load is still less than 100%, remains unaffected, thus maintaining its high QoS requirements.

A The joining/ranging process

MLAP is supported/supplemented by a MAC level management protocol (MLMP). MLMP is designed to operate independently of the underlying access protocol on a full-duplex, multi-channel, shared media, headend controlled network. Major functions of MLMP include:

- dynamic and non-disruptive (plug-and-play) join process, which requires no end-user setup;
- automatic ranging for various device parameters;
- channel assignment management to provide load balancing;
- proxy signaling for ATM circuits;
- device status and counter query;
- station removal;
- station microcode update, etc.

Although the full description of MLMP is beyond the scope of this paper, some elements of it will be briefly presented here. For more details on MLMP see [16].

The join process: The join process is the procedure that a just powered-on station uses to attach to the network. No user (application) data may be transmitted prior to the completion of this process. The join process consists of: the *invitation seeking* process; the *ranging* process; and the *station configuration* process.

During the invitation process, a newly powered-on station scans the downstream channels for an INVITATION message. INVITATION messages are sent periodically by the headend and contain one or more upstream channels on which a station can transmit an INVITATION_RESPONSE. In an INVITATION_RESPONSE message, among other information, a station provides a 48-bit IEEE address, to be replaced later by the headend with an 16-bit MLAP address. INVITATION_RESPONSE messages may require a random access scheme like the START-n algorithm to be transmitted.

During the invitation process the headend also determines the round-trip delay of data from/to a specific station. This information is used to calculate the *round-trip correction* (RTC) value to be used by the station to synchronize its upstream blocks with all the

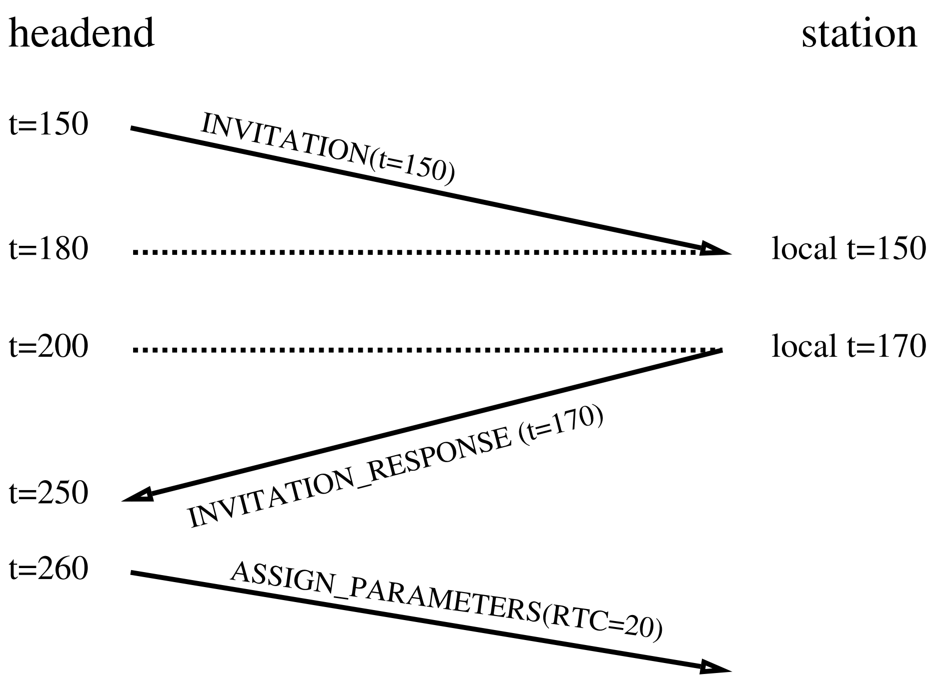

Figure 14: Calculation of the RTC value

upstream blocks of the other stations. Figure 14 illustrates an example of calculating the RTC value. The station sets its clock to the headend clock value of $t = 150$ time units. In its responds to the headend, the station provides to the headend the time that the response was sent, say, $t = 170$ time units. Note that the round-trip delay from the head-end to the station and back (or vice versa), see figure 14, is (180-150)+(250-200)=80 time units, which is also equal to the difference (250-170). Hence, as soon as the headend receives the INVITATION_RESPONSE from the station, it can immediately calculate the round-trip delay independently of the actual time that the original INVITATION message was sent. Assuming that the maximum round-trip delay, which is a known network parameter, is 100 units of time, the headend assigns the parameter RTC=20 (=100-80) units of time to the station.

The ASSIGN_PARAMETERS message also contains the MLAP address that the headend assigns to the newly powered-on station. It also includes power adjustment and equalization parameters to be used by the station transmitter.

References

[1] D. J. Aldous, "Ultimate instability of exponential back-off protocol for acknowledgement-based transmission control of random access communication channels," *Trans. Inform. Theory*, vol. 33, no. 2, pp. 219-223, March 1987.

[2] D. Bertsekas and R. Gallager, *Data Networks, 2nd ed.*, Prentice Hall, 1992.

[3] G. Fayolle, P. Flajolet, M. Hofri, and P. Jacquet, "Analysis of a stack algorithm for random multiple-access communication," *IEEE Trans. on Inform. Theory*, vol. 31, no. 2, March 1985.

[4] *IEEE Communications Magazine*, special issue on "Digital Video Communications and Video-on-Demand," vol. 32, no. 5, May 1994.

[5] *IEEE Communications Magazine*, special issue on "Reshaping Communications through Technology Convergence," vol. 33, no. 4, April 1995.

[6] *IEEE Communications Magazine*, special issue on "Access to Broadband Services," vol. 33, no. 8, August 1995.

[7] *IEEE Network*, special issue on "Digital Interactive Broadband Video Dial Tone Networks," vol. 9, no. 5, Sept./Oct. 1995.

[8] *IEEE Spectrum*, special report on "Digital TV," vol. 32, no. 4, April 1995.

[9] L. Kleinrock, *Queueing Systems, vol. 2: Computer Applications*, John Willey & Sons, 1976.

[10] J. L. Massey, "Collision-resolution algorithms for random-access communications," in *Multi-User Communications*, G. Longo (ed.), CISM Courses and Lecture Series, no. 265, Spinger-Verlag, 1981.

[11] P. Mathys and P. Flajolet, "Q-ary collision resolution algorithms in random-access systems with free or blocked channel access," *IEEE Trans. on Inform. Theory*, vol. 31, no. 2, March 1985.

[12] B. McNeil, R. Norman, and C. Bisdikian "Formal proposal for 802.14 MAC protocol (part 2 of 2): MLAP (MAC level access protocol)," contribution to the *IEEE802.14 WG* no. IEEE802.14-95/158, Nov. 1995.

[13] L. Merakos and C. Bisdikian, "Delay analysis of the n-ary stack algorithm for a random access broadcast channel," *IEEE Trans. on Inform. Theory*, vol. 34, no. 5, Sept. 1988.

[14] R. Norman, "Protocol stack and topology assumptions for MLMP and MLAP proposals," contribution to the *IEEE802.14 WG* no. IEEE802.14-95/156, Nov. 1995.

[15] R. Rom and M. Sidi, *Multiple Access Protocols: Performance and Analysis*, Springer-Verlag, 1990.

[16] R. Zeisz, "Formal proposal for 802.14 MAC protocol (part 1 of 2): MLMP (MAC level management protocol)," contribution to the *IEEE802.14 WG* no. IEEE802.14-95/157, Nov. 1995.