IEEE COMMUNICATIONS SURVEYS

ON IEEE 802.14 MEDIUM ACCESS CONTROL PROTOCOL

YING-DAR LIN
NATIONAL CHIAO TUNG UNIVERSITY, TAIWAN

ABSTRACT

In this article we present a comprehensive survey on the architecture, protocol issues, and standard of the hybrid fiber coaxial (HFC) networks which are evolving from the existing residential CATV networks. We first describe the HFC architecture and discuss the problems in providing two-way communication. Then, we identify three important medium access control (MAC) issues in designing the IEEE 802.14 standard; namely, synchronization, upstream channel access modes, and collision resolution. Resolutions adopted by the IEEE 802.14 Committee are illustrated after giving a protocol overview. Key resolutions include compensating round trip correction (RTC), interleaving minislots of data and request concurrently, and resolving collisions by multiple collision resolution engines, using the n-ary tree plus p-persistence algorithm with a first transmission rule. A comparative summary of some draft proposals that lead to the standard is given. Finally, we pinpoint two headend algorithms, minislot allocation and request scheduling, which are left open in the standard. They do not affect interoperability but may have a critical impact on performance.

We can foresee in the near future multimedia interactive services, such as video-on-demand, high quality videophone, and high-speed Internet access, will be available to residences. Both telephone and cable service providers of existing subscriber networks have a great interest in providing these services [1–4]. The CATV industry was established by providing low cost distribution of broadcast video signals to subscribers. The hybrid fiber coaxial (HFC) architecture [5, 6], in which fibers and coaxial cables are used to transport multiplexed signals to a group of 500–2000 subscribers, is becoming standard in the CATV industry and provides up to 750 MHz of bandwidth to the subscribers in the forward broadcast direction. Because the architecture does not contain switching elements in the distribution plant, and only requires optical-to-electrical and electrical-to-optical conversion, amplification, and power splitting, the plant cost is relatively low.

The HFC architecture is also being considered as a bidirectional broadband communication infrastructure, with the 5–42 MHz portion of the spectrum available in the distribution plant for transmission from the subscribers to the headend. CATV operators are interested in offering communication as well as on-demand services through this infrastructure. Thus, two standard committees, IEEE 802.14 [7–9] and Multimedia Cable Network System (MCNS) [10], were formed to seek the interoperable solutions for interactive multimedia services over the community networks.

ARCHITECTURE OF HFC

The architecture of HFC has the following important features: [5, 6]

- Tree and branch topology
- Large propagation delay
- Asymmetric upstream and downstream
- User distribution toward the tail of cable

Figure 1 represents a piece of an HFC system. A cluster of 500–2000 home subscribers are served by a fiber that comes from the headend to a fiber node. Signals are distributed to homes electrically within the serving area of a fiber node via amplified tree-and-branch feeder cables, perhaps as short as three miles in total length. One hundred twenty-five to 500 subscribers are then attached to this “last-mile” cable segment. The maximum round-trip delay between the headend and the subscriber is 0.4 msec which is equivalent to about 40 km in fiber/cable length.

Stations attached to a cable transmit and receive signal over separate frequencies. There are downstream channels (from the headend to the stations) and upstream channels (from the stations to the headend). The frequency from 50 MHz up to the upper frequency limit supported by the cable plant is allocated for downstream transmission. Within this frequency band, a channelized approach, i.e., frequency division multiplexing, with 6 or 8 MHz channels is used to transmit data from headend to stations. The sub-split band, i.e., frequency between 5–42 MHz, is allocated for upstream trans-

mission. Again, it is further divided into channels of (n x 40) kHz (n is an integer). In some cable plants, additional frequency bands for upstream transmission are intended for future use, called "mid-split" and "high-split" bands. The mid-split band extends from 5–108 MHz. The high-split covers the range between 5–174 MHz. In some geographical locations, the original sub-split band is modified as 5–42 MHz, 5–65 Mhz, and 5–48 MHz in North America, Europe, and Japan, respectively. The actual placement of channels in a given cable system depends on factors such as ingress avoidance, power allocation, etc., and shall not be specified in the standard. Digital transmission is attained by modulating the packetized digital information onto analog radio frequency (RF) carriers through 64- quadrature amplitude modulation (QAM) or 256-QAM in downstream channels and QPSK or 16-QAM in upstream channels. 64-QAM can modulate about 36 Mb/s bit stream to the downstream 6 MHz channel and quadrature phase shift keying (QPSK) offers a reliable compromise among spectral efficiency, robustness, and easy implementation to modulate, say, 320 kb/s to the upstream 160 kHz channel [8].

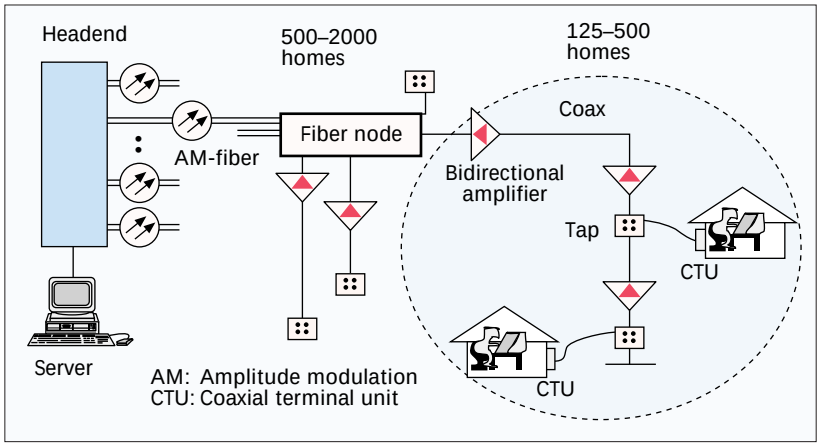

Figure 1. *The architecture of HFC.*

MAC Issues

Because stations cannot listen to the upstream transmissions of other stations directly; hence, they are incapable of detecting collisions and ultimately coordinating their transmissions by themselves. With a medium access control (MAC) protocol, stations within a branch may share the available reverse bandwidth. For the downstream channel, the problem of multiaccess does not exist because only the headend can transmit data on this broadcast channel. Part of the downstream channel bandwidth will be used to broadcast control and feedback information as required for the MAC protocol.

Three major MAC issues in the HFC networks can be identified:

- Synchronization
- Upstream channel access modes
- Collision resolution

The HFC network needs two levels of synchronization. The physical level synchronization aligns signals at the bit level and the MAC level synchronization aligns bit streams at the packet level. Here we discuss the MAC level synchronization. As the propagation delay over HFC is significant, MAC level synchronization cannot be ignored, similar to CSMA/CD of Ethernet if back-to-back frame transmissions are desired. Every station has a different propagation delay to the headend. That means when the headend reserves the time slot for station i, station i has to adapt its transmission time according to its propagation delay, so that the transmitted frame just fits in the reserved time slot when it arrives at the headend. How each station knows its propagation delay and how it utilizes the delay in frame transmission are addressed in the standard.

Since stations cannot listen to the upstream channel, they are unable to detect collisions by themselves. Aborting collided transmissions, as in CSMA/CD, to reduce the wasted bandwidth is impossible. A common practice to reduce the collided bandwidth is for the station to send a much smaller request frame to inform the headend that it needs to send a data frame. This is called reservation access mode. There is also another mode called isochronous access mode, which frees the station from sending requests periodically for the continuous traffic stream.

Still, collision may occur to small requests. We need a collision resolution mechanism. Schemes like p-persistent, binary tree walk, and n-ary tree were considered in the standard committee [11]. The n-ary tree and p-persistence resolution was adopted in the standard committee.

In the next section we give an overview the IEEE 802.14 protocol. The concepts of minislot, collision resolution interval, and access mode are illustrated. Synchronization issues are addressed in the subsequent section, followed by a description of the collision resolution mechanisms and a discussion of ways to allocation request min-slots to interleave data and requests concurrently. Draft proposals that lead to the final standard are then reviewed and compared, and in the last section we pinpoint two algorithms left open in the standard and present our conclusions.

Protocol Overview

Figure 2 is a state diagram for stations. When a station is powered up, it has to run an initialization procedure to get some network parameters. The most important aspect of the initialization procedure is ranging, which determines the round-trip correction (RTC) parameter. After the ranging process, the station can be synchronized with the headend. A dialog would take place between the station and the headend to perform a full set of registration functions.

There are possibly three access modes, namely, reservation access mode, contention access mode, and isochronous access mode. However, the standard allows only reservation and isochronous access modes due to the reason that immediate access mode wastes much bandwidth when collisions occur.

Reservation Access Mode

The reservation access mode provides the ability to dynamically assign bandwidth in a per-request manner. As its name implies, a station has to send a request telling the headend how much bandwidth it needs. The headend then reserves bandwidth for this station according to its scheduling algorithm. The concepts of minislot, piggyback, and collision resolution interval (CRI) are introduced in the standard to reduce bandwidth wastage due to collisions, to reduce the frequency of request contention, and to interleave data and requests, respectively.

Minislot — Each upstream channel is divided in time into a stream of numbered minislots. The duration of one minislot is equal to the time required to transmit eight octets of data. A MAC layer protocol data unit (PDU) that occupies

a single minislot is called a minipdu, and is used during contention opportunities for the purpose of requesting bandwidth. An integral number of minislots can be allocated by the headend and used by the station, to transmit ATM cell PDUs and variable length fragment PDUs. Assume the size of one normal data PDU is *k* times the minislot size. If a collision occurs while sending the data PDU directly, *k* times more bandwidth is wasted than if a minislot were used.

PIGGYBACK REQUEST FOR SUBSEQUENT TRANSMISSIONS — If a station has been allocated minislots to transmit its data PDUs and a new request arrives to the station, it can piggyback the request on its next transmitted PDU, instead of sending the request in a contention-based minipdu. As with a minipdu, the piggybacked request is acknowledged immediately. This mechanism means that only the first transmission of a burst has to compete for the request minislots.

REQUEST/GRANT STEPS FOR THE FIRST TRANSMISSION — Having noted the ideas of minislot and piggybacking, we now look at the request/grant steps:

(1) The station receives information from the headend on the downstream channel informing it which minislots are available to send requests.
(2) A station chooses a minislot according to the contention algorithm and places its identifier and the amount of data it wishes to send into the chosen minislot.
(3) The station waits up to a fixed amount of time until the headend sends the feedback information for its minislot transmission.
(4) If the received acknowledgment is negative, it means that a collision occurred. It then enters the collision resolution algorithm along with other stations involved in that collision.
(5) If the received acknowledgment is positive, it means that no collision occurred. The headend will later provide a grant message, to inform the station of when to transmit its data and the amount of data to transmit, or a deny message, if the access is not allowed.

ISOCHRONOUS ACCESS MODE

This access mode is intended to support constant bit rate (CBR) applications. Stations have to signal the headend to establish connections prior to data transmission. A station that wants to set up a connection first sends its request. Again, after the headend receives the request, the request is explicitly acknowledged immediately. If the headend grants this request, it constantly allocates minislots to satisfy the requested cells or minislots per second and informs the station periodically which minislot to start the transmission and how many it can transmit. A station that has established a CBR connection does not need to request again for this connection. It can send another request to end this periodical allocation and scheduling.

IMMEDIATE ACCESS MODE

The standard does not allow this access mode, although several proposals have considered this. The headend may provide unallocated upstream bandwidth to use in the immediate access mode. When the network is completely idle, in the design of some proposals, all of the upstream bandwidth is available for the immediate access mode. Usually the band-

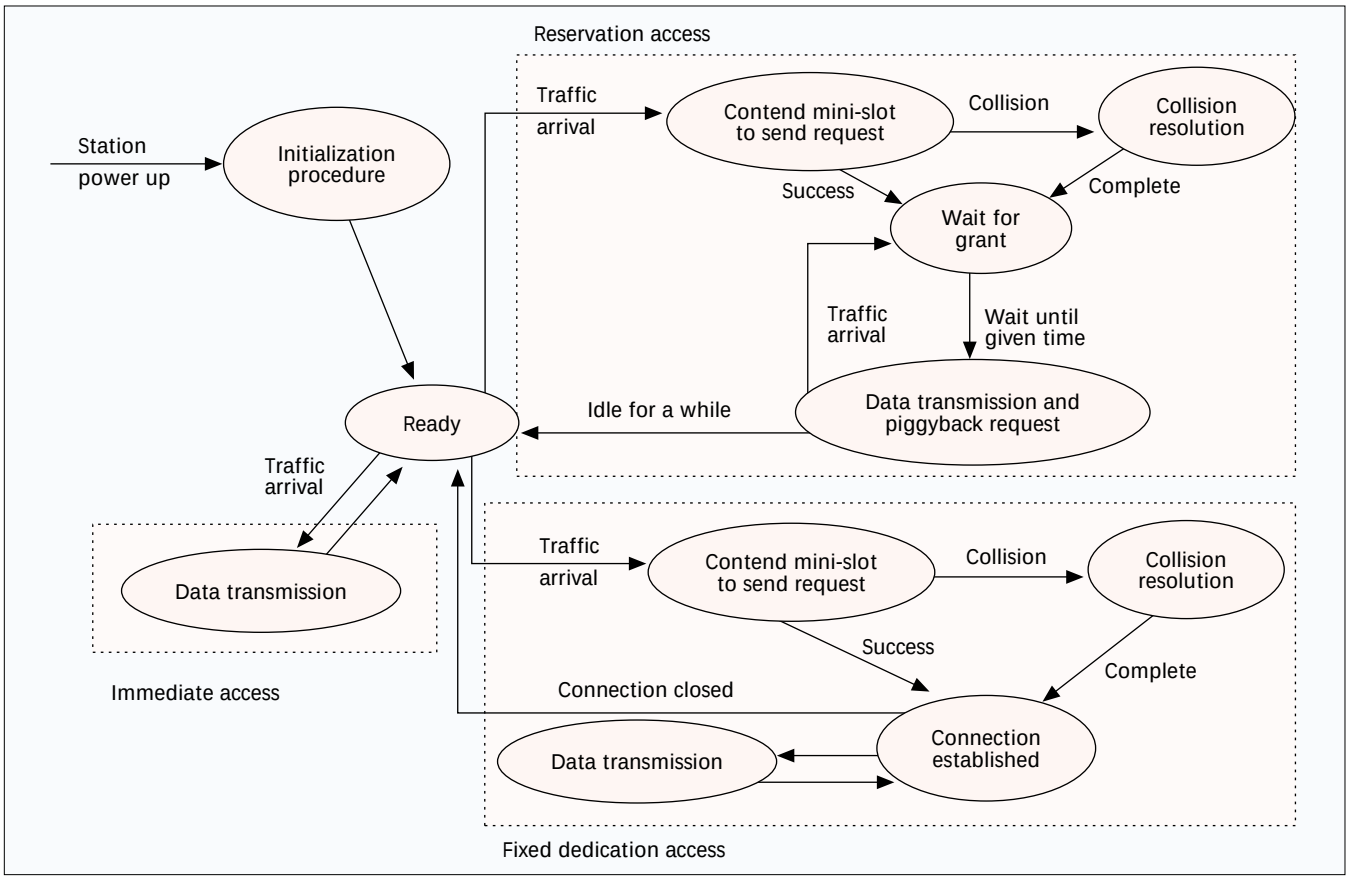

Figure 2. *Station state diagram of HFC upstream protocol.*

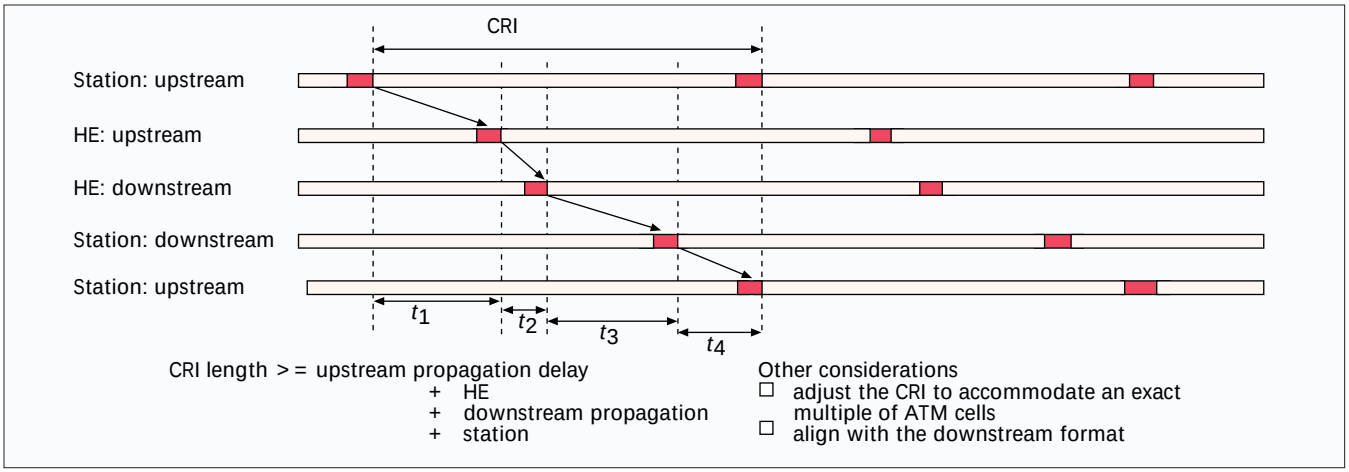

Figure 3. *Collision resolution interval: from requested transmission to acknowledgement reception.*

width is divided into small sized units for small packet transmissions and short-lived applications, for example, when a button is pressed in the interactive TV application. Large packet transmissions and long-lived applications should use reservation access or isochronous access. Since this mode is not allowed in the standard, it means no data can be sent without a previous request.

Contention Resolution Interval

A contention resolution interval (CRI) is defined as the maximum time required for all stations on the network to detect whether a request sent on a minislot was successful or collided with others. This means that a node will receive the contention result in the downstream channel before the start of the next CRI.

A CRI is shown in Fig. 3. A station first randomly chooses a minislot from the allowed ones and sends its request. After time t_1 this request arrives at the headend and the headend processing time is t_2. The headend then sends an acknowledgment, to indicate a success or collision, in the downstream channel. After time t_3 the station receives the acknowledgment and the station processing time is t_4. The CRI length must be larger than the sum of t_1, t_2, t_3, and t_4 so that in the best case, when the collision resolution algorithm allows the station to access minislots in the next cluster, the station can retransmit its request immediately. The idea of minislot cluster will be explained in detail in a subsequent section.

With the CRI idea, we can imagine that if stations send requests in an upstream channel and do not transmit anything while waiting for acknowledgments, the upstream channel is idle most of the time. That means most of the CRI time is wasted. In fact, we can interleave requests with data. That is, within a CRI, the part other than the minislot cluster can be used to transmit data PDUs. Sophisticated arrangement of data and request minislots within a CRI, including concurrent method and concurrent method with interleaving, will be described in the section "Interleaving Request with Data Minislots."

Synchronization

Compensating Network Delay

The HFC environment poses a challenging problem to the network designer since propagation delay can be much larger than the transmission time. To avoid inefficiency, every station needs to have the following information:

- Global timing reference
- Its round-trip correction

Once these are known, every station can then precisely transmit data in a given minislot assigned by the headend. With these two pieces of information, we can avoid collisions or idle periods due to different relative locations of stations.

Round-Trip Correction — During the initialization procedure, each station must perform an operation to get its RTC. An RTC value equals to the difference between the network's maximum round-trip propagation and the round-trip propagation of a station. We call this procedure ranging or station positioning. In Fig. 4, STB_A and STB_B are located at different distances from the headend. Their round-trip propagation delays are T_A and T_B. The network's maximum propagation delay is Tx. RTC_A is $Tx–T_A$ and RTC_B is $Tx–T_B$. When the headend transmits a sync message in the downstream channel, STB_A and STB_B will not see the sync message at the same time and their recognition of start of a minislot will not be the same. After the initialization procedure, each station gets its own RTC. It knows that after an RTC time it sees the sync message is the start of the minislot. In Fig. 4, if STB_A defers RTC_A and STB_B defers RTC_B after they see the sync message, both of them position themselves at the same minislot.

As illustrated in Fig. 5, the headend schedules STB_A to transmit data, starting at S_A and continuously transmit for D_A time. It means that after STB_A sees the sync message it waits one RTC_A time and then waits an extra S_A time until the minislot reserved for it. After that, it continuously transmits for D_A time. If STB_B also does the similar thing, their transmis-

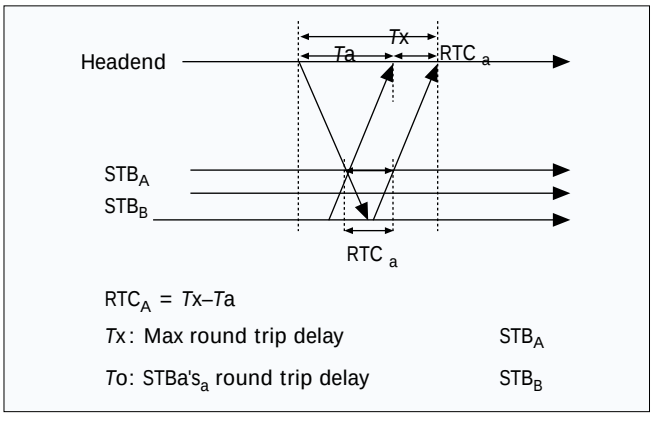

Figure 4. *Compensating network delay by deferring round-trip correction.*

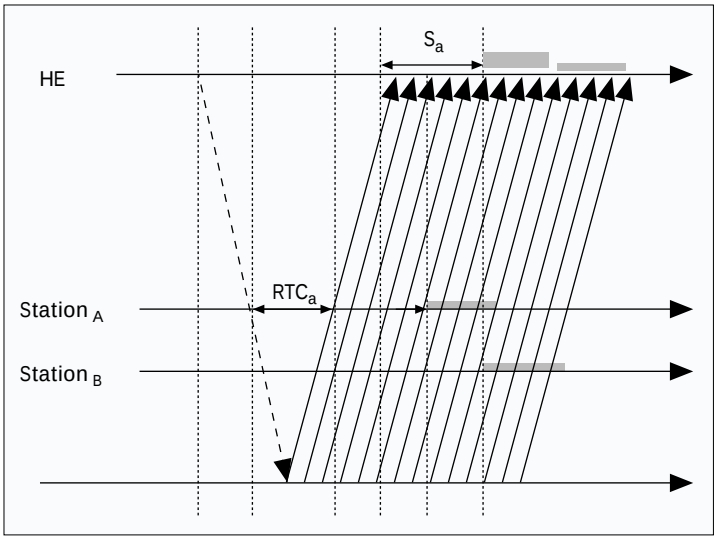

Figure 5. *Pipelined data transmission: an example with two stations.*

	Counter of A	Counter of B	Counter of C
Slot 1	0	0	0
Slot 2	random[0,2]=0	Random[0,2]=0	Random[0,2]=1
Slot 3	random[0,2]=0	Random[0,2]=2	Counter +2 = 3
Slot 4	done	Counter - 1 = 1	Counter -1 =2
Slot 5		Counter - 1 = 0	Counter -1 =1
Slot 6		Done	Counter -1= 0
Slot 7			Done

Table 1. *Updating counters to simulate the stack in the n-ary tree algorithm.*

sions will be concatenated and the channel will be pipelined with no collision or idle period.

GLOBAL TIMING REFERENCE — The headend must periodically transmit a sync message in the downstream channel. The exact interval is non-critical but is assumed to be on the order of several to tens of milliseconds. Usually we can think of the sync message as the start of a cycle. The interval from sync message to the next sync message is the length of a cycle.

RANGING

Ranging is an initialization process to measure RTC. Here we merely raise some issues. Both ranging and competition for a minislot in the reservation access mode (i.e., requesting) require contention resolution processes. The main difference between ranging in the initialization process and requesting in the reservation access mode is that the ranging process is non-slotted. In the reservation access mode each station already knows its RTC, so the upstream is synchronized. The purpose of the ranging process is to determine the RTC value. Of course the RTC value is unknown when ranging is underway.

Suppose the headend periodically invites newcomers by sending invitation messages through the downstream channel. If STBi wants to register itself to the headend, it transmits its registration request immediately when it reads the headend downstream invitation. A collision occurs when two stations, which may be on the same branch or different branches of the network, have "close" positions relative to the headend. Since most of stations are distributed around the last few miles of the cable, their propagation delays to the headend are close to each other. Thus, straightforward implementation of ranging may lead to more collisions and retries, which in turn lengthens the ranging process.

COLLISION RESOLUTION

As pointed out previously, there are two kinds of collision resolution in HFC: asynchronous collision resolution in the ranging process and synchronous collision resolution in the process of contending request minislots. In the draft standard [9], the former is resolved by p-persistence, while the latter is defined to use a hybrid of the n-ary tree algorithm and the p-persistence algorithm, with a first transmission rule. We now examine these algorithms and then illustrate how they work together in the standard.

COLLISION RESOLUTION ALGORITHMS

The theory of collision resolution algorithms for medium access in computer networks has matured for well over a quarter of a century. Many algorithms have been proposed, studied and analyzed [12–14]. There are two major categories: ALOHA-based algorithms like binary exponential backoff and p-persistence, and splitting algorithms like tree walk and n-ary tree. Each has its domain of applicability. In this section, we review the p-persistence algorithm and the n-ary tree algorithm [15] which are adopted in the standard committee.

P-PERSISTENCE ALGORITHM — In p-persistence, the headend chooses a value p ($0 < p \le 1$) and sends it to all stations. Before a station attempts a transmission, a number is randomly generated between zero and one. If this value is smaller than p, the station transmits; otherwise the station waits. One feature of this algorithm is that whether or not to transmit is independent of previous attempts. No state is stored in the station and the headend.

The efficiency of this algorithm depends on the value p. The optimal value of p can be proved to be $1/N$, where the N is the number of stations involved [14]. One optimization of this algorithm is to adjust p every time there is a successful transmission. We refer to it as adjusted p-persistence. Therefore, the value of p in the adjusted p-persistence algorithm corresponds to $p = 1/n$, where n follows the sequence $\{N, N–1, N–2, ...,2,1\}$. That means each time the headend successfully receives a request, the number of stations involved in the collision is decreased and the headend announces the new p value via the downstream channel.

N-ARY TREE ALGORITHM — In this algorithm, each station has a counter (Table 1). The counter value indicates how many *slots* you have to let pass before your next transmission. If the counter equals zero, the station is permitted to transmit in the next slot. Depending on the result of the contention slot, the counter of each station is adjusted as follows:

Collision:
- If the station is involved in this contention slot, counter = random[0, n–1].
- If the station is not involved, counter = counter + (n–1).
- Note that n is the branching degree of the n-ary tree algorithm.

No collision (success or idle):
- If the station is involved in this contention slot, it has successfully transmitted its message.
- If the station is not involved, counter = counter–1.

We can imagine that there is a virtual stack. The above rules mean that the stations involved in a contention should randomly select a position in the virtual stack from zero to *n*–1. The other stations that are already in the stack (i.e., not involved in that contention) lower their positions by one if no collision occurs and raise their positions by *n*–1 if a collision occurs.

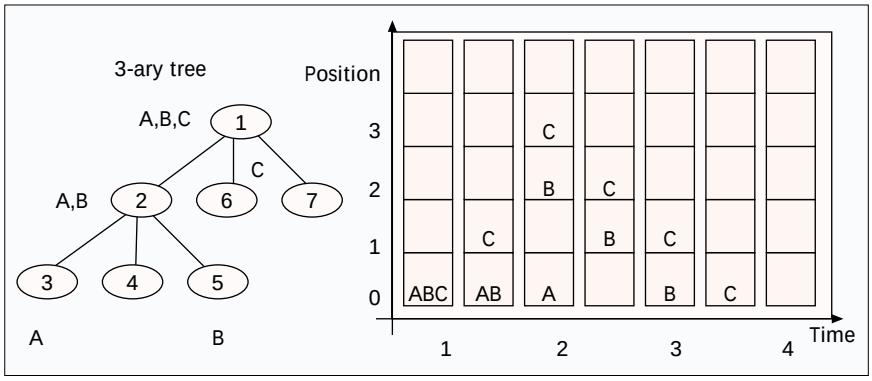

Figure 6. *Operation of the n-ary tree algorithm: tree and stack,* n *= 3.*

The n-ary tree algorithm can be divided into the following two cases:
- Blocking: some implementations block newcomers from joining the collision resolution. That means newcomers are not allowed to transmit their requests until the current contention is fully resolved. Newcomers can only join a new contention when the virtual stack is empty. These are usually referred as tree-search algorithms.
- Non-blocking: in non-blocking implementations, each newcomer has a counter value zero. Thus it can join the collision resolution any time, even when the collision resolution is ongoing. These are usually referred as stack algorithms.s

Figures 6 and 7 show an example run of this algorithm with $n = 3$. The stack position in Fig. 6 is represented by the counter in Fig. 7. In the beginning, stations A, B, and C contend for slot 1, which leads to a collision. Then A, B, and C randomly choose a backoff time between 0 and 2 (i.e., *n*–1). Stations A and B choose 0 and station C chooses 1. Now the positions of A, B, and C in the stack are 0, 0, and 1, respectively. A and B collide again at slot 2. Three slots are allocated to resolve this collision. C's position in the stack is increased by 2, i.e., *n*–1. A and B choose 0 and 2, and successfully transmit their messages at slots 3 and 5, respectively, while leaving slot 4 idle. C's position is decreased by 1 at each non-collided slot. Finally it reaches position 0 and transmits successfully.

Suppose there is a newcomer D who wants to join the collision resolution at slot 2. In a blocking implementation, D cannot transmit its request until slot 8. In a non-blocking implementation, D can join the collision resolution right away.

The proposal from IBM, MLAP [16–18], uses this algorithm with $n = 3$ and can switch between non-blocking and blocking modes. Scientific Atlanta's XDQRAP [19] uses this algorithm with $n = 2$ and blocking mode. The simulation result from [11] shows that the ternary ($n = 3$) tree algorithm achieves the shortest collision resolution interval and the binary ($n = 2$) tree algorithm is close behind.

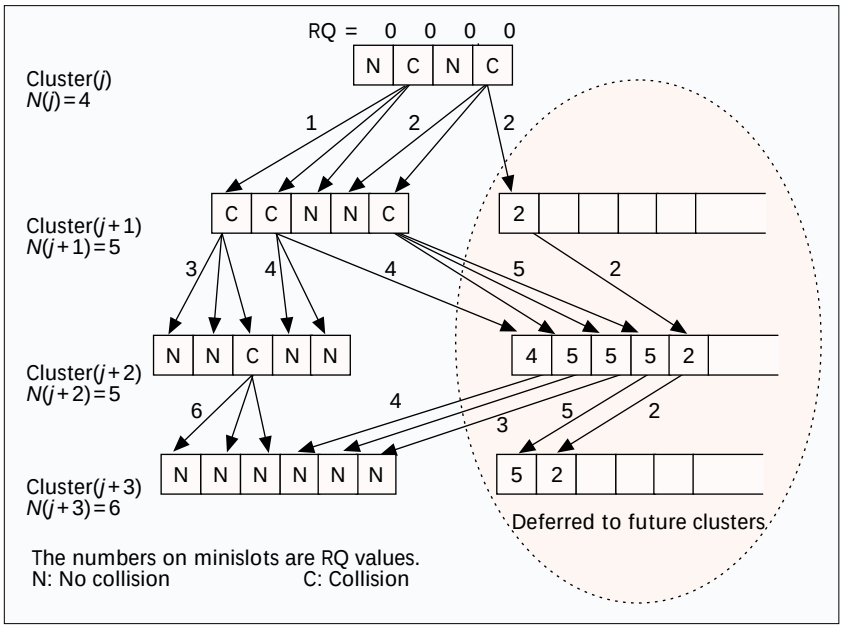

Figure 7. *Operation of the ternary tree algorithm with first transmissions and retransmissions.*

Collision Resolution in IEEE 802.14

There is consensus in the IEEE 802.14 committee [9] that the standard collision resolution algorithm is a fairly complicated combination of the priority plus FIFO first transmission rule, the n-ary tree plus p-persistence collision resolution algorithm, and multiple collision resolution engines running in parallel.

Minislot Allocation Algorithm — The headend runs a minislot allocation algorithm to decide the number of minislots in a cluster for the purpose of sending requests. The standard leaves this open. The algorithm can be smart enough to allocate more minislots when the virtual stack goes high and allocate fewer minislots when the stack goes low.

From time to time, the headend sends an allocation map describing the location and usage of a minislot cluster, i.e., a block of contiguous request minislots. Specifically, an allocation map PDU indicates the collision resolution engine this map comes from, the number of the first request minislot this map specifies, and the division of the request minislot cluster into groups with different resolution queue (RQ) values. The group whose RQ is zero is further divided into subgroups with different priorities and admission time boundaries, which are used to enforce the priority+FIFO first transmission rule. The other groups, with RQ greater than zero, are used for retransmissions. The descriptor for any of them specifies the number of allocated minislots, i.e., *n* of the n-ary tree algorithm, and the split value, which is used to run p-persistence where *p* is equal to *n* divided by the split value. Note that the values in descriptors can vary from one group to another.

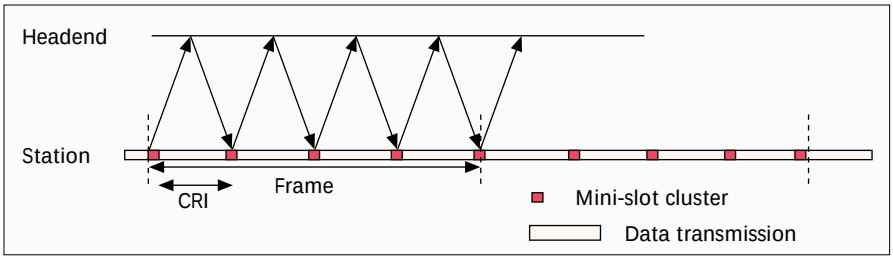

Figure 8. *Layout of concurrent method.*

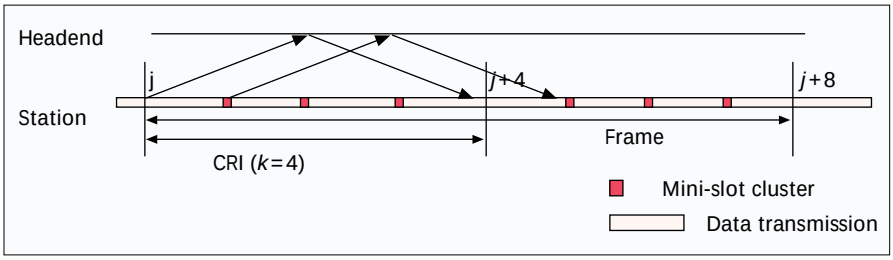

Figure 9. *Layout of concurrent method with interleaving (frame size =2 * CRI).*

FIRST TRANSMISSION RULE: PRIORITY AND FIFO — A newly arriving request can only use the request minislots of the group with RQ value equal to zero. The request can only use the subgroup whose priority is the same as its priority. If a match does not exist, the request is blocked and retried when the next allocation map comes. If a match is found, the admission time boundary of the subgroup is compared with the arrival time of the request. Only when the arrival time is smaller, i.e., earlier, than the admission time boundary can the request be transmitted in a randomly selected minislot of the subgroup. That is, the request must be old enough to get transmitted.

RETRANSMISSION RULE: N-ARY TREE AND P-PERSISTENCE — If a collision has occurred to a transmitted request, the station, which sent the request, saves the RQ value provided to it in the headend acknowledgment message. When the headend announces the arrival of the next minislot cluster by an allocation map, the station checks the RQ values of the groups in the cluster to see if any of them is smaller than or equal to the saved RQ value. If none exists, the station continues to wait. If one or more than one group is found, the station will try to contend in the group, among the eligible groups, with the largest RQ value. The station now randomly selects a number from 1 to the split value. If the number, for example *m*, is smaller than or equal to the number of allocated minislots of the group, it transmits the request on the *m*th minislot of the group; otherwise, it waits for the next allocation map and retries. The random selection step is the n-ary tree algorithm, while the final check is the p-persistence algorithm.

As shown in Fig. 7, newcomers randomly pick a number, *i*, between 1 and 8, the split value, and there are four minislots in the group, within cluster *j*, with RQ equal to 0. First and third minislots are either successful or idle, i.e., [N]. Collision occurs, i.e., [C], for those stations with *i* equal to 2 or 4. Those stations with *i* equal to 5, 6, 7, and 8 do not transmit anything in this cluster, but will retry in the next available cluster. For those stations with *i* equal to 2 (i.e., a collision), the headend provides an acknowledgment with RQ = 1. These stations save this RQ value and when the next cluster of minislots is announced, it checks the RQ values to see if any of them is smaller than or equal to the saved RQ value. If it is, the station contends; otherwise, it waits. Simulation results show this can achieve a better performance than the pure ternary tree algorithm [19].

INTERLEAVING REQUEST AND DATA MINISLOTS

Now we are ready to describe the sophisticated arrangement of request and data minislots within a CRI.

CONCURRENT METHOD

The concurrent method overlaps the collision resolution phase and the data transmission phase. While a station waits for the acknowledgment of a previous request, another upstream data transmission is under way. The data transmission part and request minislot part are mixed, as shown in Fig. 8. In this method, a frame contains several clusters of minislots and data transmission parts. In a CRI, each cluster of minislots is followed by the data transmission part. After sending a request in cluster *j*, a station knows whether its request is successfully transmitted or not before cluster *j*+1 comes. In the best condition, the station can retransmit its request right away in cluster *j*+1 if the collision resolution mechanism allows.

CONCURRENT METHOD WITH INTERLEAVING

The concurrent method with interleaving has multiple contention resolution engines. Each contention resolution engine operates independently (i.e., interleaving). In the concurrent method we discussed above, there is only one minislot cluster per CRI. With interleaving, we can have more than one minislot cluster per CRI, as shown in Fig. 9. For a network with a length of 40 km, the CRI becomes 400 μs, which not only increases the collision resolution cycle and lengthens the waiting time of the blocked requests. We can use interleaving to decrease the waiting time of the blocked requests. But using multiple collision resolution engines requires to maintain multiple queues in the headend. This will increase the headend complexity. We may instead use a good minislot allocation algorithm with a variable number of minislots to satisfy a network of large propagation delay, without increased complexity.

If there are *k* clusters of minislots in a CRI, the interleaving factor for the network becomes *k*. There are *k* contention resolution engines running in parallel. For example, clusters {1, *k*+1, 2*k*+1,...} are controlled by the same collision resolu-

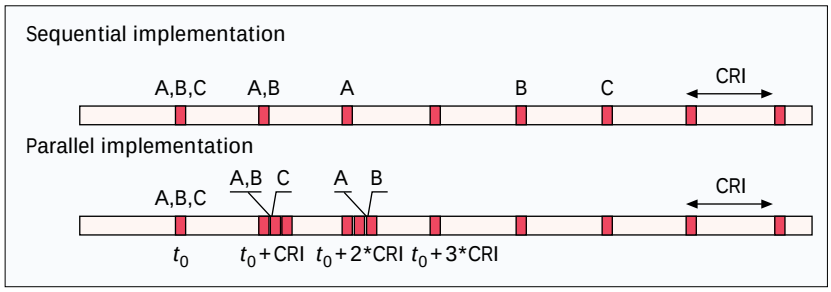

Figure 10. *Examples of sequential implementation and parallel implementation.*

tion engine. In this method, a frame consists of several CRIs. After sending a request in cluster *j*, a station can know whether its request is successfully transmitted or not before cluster $j+k$. In the best condition, the station can retransmit its request in cluster $j+k$ if the collision resolution mechanism allows.

Sequential and Parallel Implementations

Among the IEEE 802.14 proposals one can identify two main classes of implementation, namely sequential implementation and parallel implementation. If a collision occurs at time t_0, then at t_0 + CRI each station involved in this collision sets its counter randomly between 0 and 2 (i.e., n–1). Now we use the procedure we described above for the n-ary tree algorithm to adjust the counter value, but with two different ways to allocate minislots:

Sequential implementation
- Minislot clusters are arranged at $\{t_0, t_0 + \text{CRI}, t_0 + 2*\text{CRI},...\}$. Each minislot cluster has only one minislot.

Parallel implementation
- Minislot clusters are arranged at $\{t_0, t_0 + \text{CRI}, t_0 + 2*\text{CRI},...\}$. Each minislot cluster may have multiple minislots.

We use the same example as in Fig. 6 to explain the difference between these two implementations, as shown in Fig. 10. When a collision occurs at time t_0, in the sequential implementation minislots are arranged at $\{t_0+\text{CRI}, t_0+2*\text{CRI}, t_0+3*\text{CRI}\}$ and in the parallel implementation minislots are arranged at $\{t_0+\text{CRI}, t_0+\text{CRI}+1, t_0+\text{CRI}+2\}$. The sequential implementation takes seven CRI times to resolve the collision occurs at t_0. The parallel implementation only takes three CRI times. The sequential implementation is simple; however, the parallel implementation can achieve a shorter collision resolution time. There are also many other possible parallel implementations, but some of them lead to nontrivial minislot allocation.

The IEEE 802.14 committee does not include the minislot allocation algorithm, although some suggestions are given. This does not affect the interoperability because the headend periodically announces, in the downstream channel, where the contention minislots are. Combining the ideas of concurrent minislot cluster and interleaving, Table 2 summarizes four combinations of strategies in allocating request minislots. For example, since the concurrent method has a minislot cluster per CRI (i.e., $k = 1$), if we use a sequential implementation, each CRI in turn has only one minislot.

	Sequential	Parallel
Concurrent ($k=1$)	One cluster per CRI One minislot per cluster	One cluster per CRI More than one minislot per CRI
Concurrent with interleaving ($k>1$)	More than one cluster per CRI One minislot per cluster	More than one cluster per CRI More than one minislot per cluster

Table 2. *Sequential or parallel implementation of concurrent minislot cluster with or without interleaving.*

Proposal	XDQRAP	MLAP	ARAP	FPP	802.14
Proposed by	Scientific Atlanta	IBM	Zenith	NEC	IEEE
Frame layout	Concurrent	Concurrent	Concurrent	X	Concurrent
Piggyback	No	Yes	Yes	Yes	Yes
Multiple priority	No	Yes	No	No	Yes
ATM support	Yes	Yes	Yes	No	Yes
Collision resolution	Binary tree	Ternary tree (START-3)	p-persistence	X	Ternary tree + p-persistence
Blocking	Yes	No	Yes	X	Yes
Interleaving	Yes	No	No	X	Yes
Number of minislots	Fixed	Fixed	Variable	X	Variable

X: FPP has no such features.

Table 3. *Feature comparison of MAC proposals.*

	XDQRAP	MLAP	ARAP
Max throughput (BW=3 Mb/s)	1.705 Mb/s	1.77408 Mb/s	2.21 Mb/s
Mean access delay lower bound	2.521 ms	2.33 ms	6.6 ms

Table 4. *Maximum throughput and lower bound of mean access delay: XDQRAP, MLAP, and ARAP.*

Proposals Review

Table 3 is a comparative summary of several IEEE 802.14 proposals. FPP [20] is a polling protocol and XDQRAP [19], MLAP [16–18], and ARAP [21] are request/grant protocols. The main problem with FPP is the need to maintain an online station table.

Since the FPP key features are not included in the draft standard, we only discuss XDQRAP, MLAP, and ARAP here. The simulation results in Table 4 are from [22]. We use these simulation results to explain the influence of the characteristics we discussed. The detailed simulation model and parameters are not mentioned here.
- Since the number of minislots per cluster remains fixed in XDQRAP, it has more collisions and longer collision resolution time. In supporting VBR traffic, it does not piggyback the request, which is inefficient. And in supporting CBR traffic, its fixed small size data slots cause a lot of packet segmentation. Its throughput is slightly lower.
- MLAP uses its own 3-ary tree collision resolution algorithm, START-3. It is a very efficient collision resolution algorithm but it only allocates a fixed number of minislots per cluster. If it can support a variable number of minislots, the collision resolution mechanism may have better performance. Another feature in MLAP is that the short bursty traffic may have lower access delay because it supports multiple priorities.
- In ARAP a larger frame size is used, which leads to a

higher access delay under a light offered load. However, due to a variable number of minislots per cluster, the number of collisions is lower and the collision resolution time is much shorter. ARAP can achieve a lower access delay under a high offered load.

The standard is not just one of the protocols but a hybrid of them. Many of the proposals brought forward have similar defining characteristics. All of these good characteristics are put into the standard. For example, IEEE 802.14 uses a n-ary tree collision resolution with soft blocking. That is a combination of ARAP's p-persistence based algorithm and MLAP's START-3 algorithm. The idea of supporting a variable number of minislots per cluster is from ARAP. And the idea of interleaving comes from XDQRAP.

Conclusion and Future Work

This paper presents a survey of the medium access control protocol in the HFC network. We identify three important issues: synchronization, collision resolution, and the layout of collision resolution interval (CRI). We compare alternatives and illustrated IEEE 802.14's solution.

The conclusion is that IEEE 802.14 has the following properties:

- Support both fixed sized ATM cells and variable length packets
- Support reservation access and isochronous access
- Provide concurrent data transmission and request collision resolution
- Exercise an n-ary tree based collision resolution mechanism, by allocating a fixed or variable number of minislots, where the allocation can be implemented sequentially or parallelly and possibly by multiple collision resolution engines

The IEEE 802.14 standard is being released. There are several algorithms that the standard does not and will not cover. Future research work can be continued in these directions:

- The request scheduling algorithm — When the headend receives bandwidth requests, how should it schedule the stations' data transmission?
- The minislot allocation algorithm — In a parallel implementation, how many minislots should be there in a minislot cluster?

Some request scheduling algorithms can be found in [6] and [23]; the former enforces a fair cyclic scheduling discipline while the latter adopts a priority on-the-fly discipline. None of them address the minislot allocation problem. Since request minislots occupy the upstream minislots, how they are allocated affect the number of minislots available to the scheduling algorithm. Thus, the interdependency of these two algorithms should be further studied.

Acknowledgments

The author would like to thank the anonymous reviewers for the detailed reviews on an earlier manuscript. Their comments have improved the presentation of this paper. Also, thanks go to Chun-Hong Lee who provided much input for the material reported here.

References

[1] Leif A. Ims, Dagfinn Myhrem, and Borgar T. Olsen, "Economics of Residential Broadband Access Network Technology and Strategies," *IEEE Network,* Jan./Feb. 1997, pp. 51–57.
[2] Chatschik Bisdikian, Kiyoshi Maruyama, David I. Seidman, and Dimitrios N. Serpanos, "Cable Access Beyond the Hype: On Residential Broadband Data Service over HFC Networks," *IEEE Commun. Mag.,* Nov. 1996, pp. 128–135.
[3] Marlis Humphrey and John Freeman, "How xDSL Supports Broadband Services to the Home," *IEEE Network,* Jan./Feb. 1997, pp. 14–23.
[4] William Pugh and Gerald Boyer, "Broadband Access: Comparing Alternatives," *IEEE Commun. Mag.,* Aug. 1995, pp. 34–46.
[5] Charles A. Eldering, Nageen Himayat, and Floyd M. Gardner, "CATV Return Path Characterization for Reliable Communications," *IEEE Commun. Mag.,* Aug. 1995.
[6] Ying-Dar Lin, Chia-Jen Wu, and Wei-Ming Yin "PCUP : Pipelined Cyclic Upstream Protocol Over HFC," *IEEE Network,* vol. 11, no. 1, Jan./Feb. 1997, pp. 24–34.
[7] John W. Eng and James F. Mollenauer, "IEEE Project 802.14: Standards for Digital Convergence," *IEEE Commun. Mag.,* vol. 33, no. 5, May 1995, pp. 20–23.
[8] IEEE, "IEEE Project 802.14/a Draft 2 Revision 2: Cable-TV Access Method and Physical Layer Specification (status: interim)," July 1997, http://www.walkingdog.com, http://www.802.14.org
[9] IEEE, "IEEE Project 802.14/a Draft 3 Revision 1: Cable-TV Access Method and Physical Layer Specification (status: interim)," April 1998, URL: http://www.walkingdog.com, http://www.802.14.org
[10] Cable Television Laboratory Inc., "Data-over-Cable Service Interface Specification: Radio Frequency Interface Specification (draft 2 revision 2, status: interim)," Oct. 8, 1997, http://www.cablemodem.com
[11] Chatschik Bisdikian, "A Review of Random Access Algorithms," contribution to the IEEE 802.14 WG, no. IEEE 802.14 -96/019, January 1996.
[12] IEEE Transactions on Information Theory, special issue on "Random Access Communications," March 1985.
[13] Norman Abramson, Ed., "Multiple Access Communications: Foundations for Emerging Technologies," IEEE Press, 1993.
[14] Andrew S. Tanenbaum, "Computer Networks, Third Ed.," Prentice-Hall, 1996, pp. 246–266.
[15] P. Mathys and P. Flajolet, "Q-ary Collision Resolution Algorithms in Random Access Systems with Free or Blocked Channel Access," IEEE Transactions on Information Theory, March 1985.
[16] Ray Zeisz, "Formal Proposal for 802.14 MAC Protocol (Part 1 of 2): MLMP (MAC Level Management Protocol)," contribution to the IEEE 802.14 WG, no. IEEE 802.14-95/157, Nov. 7, 1995.
[17] Bill McNeil, Rob Norman, and Chatschik Bisdikian, "Formal Proposal for 802.14 MAC Protocol (Part 2 of 2): MLAP (MAC Level Access Protocol)," contribution to the IEEE 802.14 WG, no. IEEE 802.14-95/158, November 7, 1995.
[18] Chatschik Bisdikian, "MLAP: A MAC Level Access Protocol for the HFC 802.14 Network," *IEEE Commun. Mag.,* vol. 34, no. 3, Mar. 1996, pp. 114–121.
[19] Frank Koperda, Bouchung Lin, and Jason Collins, "A Proposal to Use XDQRAP for 802.14," contribution to the IEEE 802.14 WG, no. IEEE 802.14-95/068, July 12, 1995.
[20] Morihisa Momona and Shuntaro Yamazaki, "Framed Pipeline Polling for Cable TV Network," contribution to the IEEE 802.14 WG, March 1, 1995.
21] Richard Citta and David Lin, "Formal MAC Layer Proposal: Adaptive Random Access Protocol for CATV," contribution to the IEEE 802.14 WG, no. IEEE 802.14-95/144, Oct. 23,1995.
[22] David H. Su *et al.*, "Preliminary Simulation Results of the MAC Protocol Proposals," contribution to the IEEE 802.14WG, no.IEEE-96/126, May 1996.
[23] John O. Limb and Dolors Sala, "A Protocol for Efficient Transfer of Data Over Hybrid Fiber/Coax Systems," *IEEE/ACM Trans. on Networking,* vol. 5, no. 6, Dec. 1997, pp. 872–881.

Biography

Ying-Dar Lin (ydlin@cis.nctu.edu.tw) is an associate professor in the Department of Computer and Information Science at National Chiao Tung University, Hsinchu, Taiwan. His research interests include design and analysis of high-speed LANs/WANs, broadband access networks, wirelss networks, ATM and IP switching networks, and network-centric computing. He received a Bachelor's degree in computer science and information engineering from National Taiwan University in 1988, and M.S. and Ph.D. degrees in computer science from the University of California, Los Angeles in 1990 and 1993, respectively. He joined National Chiao Tung University in 1993. He is a member of ACM and IEEE.