COMMUNICATING WITH VIRTUAL PATHS AND VIRTUAL CHANNELS

Rick Bubenik Mike Gaddis John DeHart

Applied Research Laboratory
Department of Computer Science
Washington University
St. Louis, Missouri 63130-4899

ABSTRACT We compare *virtual path* and *virtual channel* connections in ATM (Asynchronous Transfer Mode) networks. Both types of connection can be used to support one-to-one, one-to-many, many-to-one, and many-to-many communication between clients of the network. We give several examples of their use for these types of communication, then compare the two mechanisms. For most types of communication, both virtual path and virtual channel connections are more-or-less equally capable. However, for relatively high bandwidth communication involving more than five to ten participants, virtual path connections can more efficiently support some interconnection patterns. In general, we believe that both connection types are desirable and both should be extended to the client by network providers.

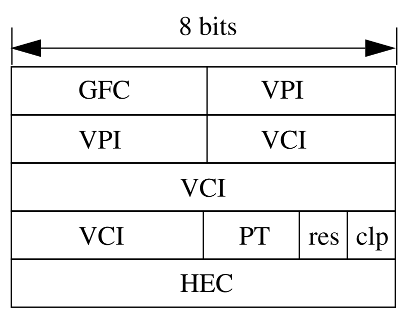
1 Introduction

Clients of ATM (Asynchronous Transfer Mode) networks communicate by creating *connections* to one another, then exchanging data over these connections [1] [2] [4] [6] [7] [10]. The ATM standard provides for two types of connections: *Virtual Path* (VP) and *Virtual Channel* (VC). With a VP connection, clients set the VP identifier (VPI) field of the ATM header (Figure 1) when sending cells. The network then uses the VPI for routing, possibly remapping this field at every switching node within the network, until the cells reach their destinations. With VC connections, clients set the VC identifier (VCI) and the VPI when sending cells and the network uses both the VCI and VPI for routing. For VP connections, the VCI is preserved by the network—whatever value the sending client places in this field is delivered to the destination client(s) and is available for use by the client, for example, as a multiplexing field. With VC connections, the VPI is not necessarily preserved by the network and may have to be set to a particular value (such as zero). Therefore, VC connections do not allow the client to use the ATM header for multiplexing.

VP and VC connections can be used to support: 1) one-to-one communication involving two network clients, useful for traditional data exchange, 2) one-to-many communication, where one transmitter sends data to many receivers, useful for broadcast video distribution, 3) many-to-one communication, where several transmitters send to one receiver, useful for error logging and monitoring, and 4) many-to-many communication, where all clients send to and receive from one another, useful for video conference calls and distributed processing applications [5].

In order to compare the strengths and weaknesses of VP and VC connections in supporting these types of communication, we first define some terminology. We define an *ATM network connection* to be a single end-to-end VP or VC connection created by the network, usually as a result of a signaling message sent from a client to the network. In general, an ATM network connection can have any number of transmitters and receivers. A *unidirectional* ATM network connection is one with only one transmitter. We define a *logical channel* to be a communication channel over which two or more clients of the network communicate. When one client sends a message out over a logical channel, all other clients on the logical channel receive copies of the message (reliability can be implemented, but is not implied). By this definition, a logical channel can be one-to-one, one-to-many, many-to-one, or many-to-many. Logical channels can be realized in three different ways: 1) a logical channel may have a one-to-one correspondence with an ATM network connection, 2) several logical channels may be mapped to the same ATM network connection, or 3) multiple ATM network connections may be used to implement a single logical channel.

The choice as to which type of connection, VP or VC, is preferable depends largely on the manner in which the network clients are communicating. If the clients use multiple logical



GFC—Generic Flow Control
VPI—Virtual Path Identifier
VCI—Virtual Channel Identifier
PT—Payload Type
res—Reserved
clp—Cell Loss Priority
HEC—Header Error Check

Figure 1. ATM User Network Interface (UNI) Header Format (5 bytes).

From: *INFOCOM '92: Proceedings of the Eleventh Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 1035-1042, May 1992.

channels and/or multicast communication, a means is required to differentiate between cells arriving on separate logical channels and from different transmitters. With VP connections, the VCI can be used for this purpose. With VC connections, an identifier embedded in the payload can be used for differentiation, or separate VC connections can be used for this purpose. If an identifier is embedded in each cell, the amount of payload field usable by higher level protocols is reduced. If separate VC connections are used, the VCI that cells arrive on can be used for differentiation, but bandwidth sharing between the separate connections is not possible in many networks, leading to higher bandwidth requirements and a higher probability of connection blocking (Section 3). Alternatively, a higher level protocol, per frame identifier can be used for differentiation (Section 2.1). With this strategy, intermixing of cells from different frames makes reassembly difficult, if not impossible.

We elaborate on the manner in which VP and VC connections can be used in Section 2. In Section 3, we discuss bandwidth sharing and the way in which the different communication options affect the ability to share. In Section 4, we give an overview of a simple host interface that supports both VP and VC connections. Finally, in Section 5, we summarize.

2 Multiplexing and Demultiplexing Strategy

Because most applications communicate in larger sized data units than the small 53 byte ATM cell size, data that is transmitted over ATM networks will typically be segmented before transmission and reassembled after reception. Several segmentation and reassembly schemes have been devised, some of which are currently being standardized [1] [4] and others of which have been proposed for standardization [3] [9]. A scheme for segmenting and reassembling IP datagrams over an ATM network is discussed by Lyon [9], and shown pictorially in Figure 2. With this scheme, each frame (datagram) is segmented into a sequence of cells, where the last cell is marked (using the PT field) as an *end* cell. Since the ATM standard requires that cells not be reordered while in the network, reassembly based on these designations is straightforward. To add protection against lost and reordered cells, a CRC is appended to the end of the frame, where reassembled frames are discarded if the CRC is incorrect. To make the reassembly procedure more robust in the face of cell loss, *start*, *middle*, and *end* designations can be used instead of a single *end* designation [3].

This simple segmentation and reassembly procedure works well, if only two clients are communicating over a single logical channel. However, if two clients wish to use more than one logical channel or if additional clients are added to the connection, a means of differentiating the logical channels and/or the transmissions of each client is required. We have identified three solutions to this problem. The first is to transmit all data over a single ATM network connection and use per frame information (such as the source address and port in the transport header) for differentiation (Section 2.1). This solution only works if cells from different frames do not intermix. The second solution is to transmit all data over a single ATM network connection and use per cell identifiers for differentiation (Section 2.2). This allows cells to become intermixed and still be differentiable. The third solution is to transmit over several ATM network connections, using the different connections for differentiation (Section 2.3). This solution also allows for cell intermixing on the link, although intermixing on each connection is avoided. Hybrids of these approaches are also possible (Section 2.4).

2.1 Method 1: Use of Per Frame Identifiers

2.1.1 Point-to-point Connections

When two clients communicate over a point-to-point connection using more than one logical channel, the TCP port (or its equivalent in other protocols) of each reassembled frame can be used to determine which logical channel each frame is associated with. The clients must avoid intermixing the segmented cells from separate ports since, if intermixed, there is no way to differentiate the cells from one port with those of another. If intermixing occurs, upon reassembly, the CRCs will be incorrect and the reassembled frames will be discarded.

2.1.2 Multipoint Connections

With multipoint connections having more than two transmitters, the IP source address (or its equivalent) of the reassembled frame can be used to determine the transmitter and the TCP port can be used to determine the logical channel. However, if two transmitters send segmented frames simultaneously, it is likely that the cells from each stream will intermix, causing the reassembled frames to be discarded due to CRC errors. Additionally, the streams may intermix at some clients and not others,

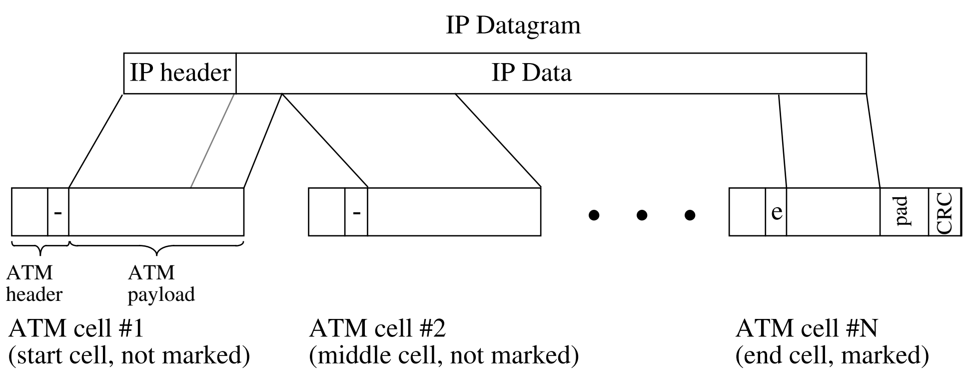


Figure 2. IP Segmentation/Reassembly Scheme.

resulting in selective loss. If the probability of two or more simultaneous transmissions is low enough, the frame loss due to cell intermixing will be infrequent and higher level protocol frame retransmissions can mask the problem without serious performance degradation.

To calculate the likelihood of cell intermixing, consider a situation where N transmitters share a multipoint connection. Assuming that all transmitters are independent, the pairing of each transmitter and its access link can be modelled as an M/M/1 queue, where the probability that a transmitter is active is given by ρ. The probability that two or more transmitters are active is one minus the probability that either no transmitters are active or exactly one transmitter is active, given by:

$$1 - [(1-\rho)^N + N\rho(1-\rho)^{(N-1)}]$$

This probability is the worst case probability of cell intermixing. In practice, cell intermixing may not always occur when two or more transmitters are active because of propagation delays in the network. For example, when the first cell of one transmission enters the network while the last cell of another is leaving, intermixing will likely be avoided.

Using this upper bound, Table 1 shows the probability of cell intermixing for various values of N and ρ (all entries are rounded to two decimal places). Assuming a 150 Megabit link, the bandwidth percentages correspond to usage in the range of approximately 1.9 Kbytes/second (at 0.0001) to 1.9 Mbytes/second (at 0.1), including header overhead. With 25 or less transmitters, performance appears acceptable up to 190 Kbytes/second. With more transmitters or higher bandwidth usage, performance degrades to unacceptably low levels. This type of connection does not appear suitable for a file server/workstation client configuration, but would work as a common *signaling* channel (used for setting up other connections, as discussed in Section 2.2.2) or for other low-to-moderate bandwidth communication (for example, ARP, the Internet Address Resolution Protocol).

2.2 Method 2: Use of Per Cell Identifiers

The second method for multiplexing and demultiplexing multiple logical channels and multiple transmitters is to send all data on a common ATM network connection and use per cell

Bandwidth \ Number of transmitters	10	25	50	100	150	200	300	500
0.0001	0	0	0	0	0	0	0	0
0.001	0	0	0	0	.01	.02	.04	.09
0.01	0	.03	.09	.26	—	—	—	—
0.1	.26	—	—	—	—	—	—	—

Table 1. Cell Intermixing Probability on a Multicast Connection as a Function of Normalized Bandwidth and Number of Transmitters.

identifiers for differentiation. If clients are communicating with a VP connection, the VCI can be used for differentiation. Otherwise, an identifier can be embedded in the ATM payload field of each cell, such as the type 4 ATM Adaptation Layer (AAL) Message Identifier (MID) field [1] [4]. The techniques that clients use to negotiate unique identifiers are the same regardless of where the identifier is located. In the remainder of this subsection, we assume a VP connection is employed and describe the techniques for assigning VCIs. Analogous techniques can be used to assign embedded identifiers, such as the MID.

The primary advantage of using a per cell identifier over use of a per frame identifier (Section 2.1) is that the cells from different logical channels and from different transmitters can intermix on the link and still be separable, allowing the intermixed cells to be reassembled.

2.2.1 Point-to-point Connections

When two clients communicate over a point-to-point connection using multiple logical channels, a means is needed to distinguish between the multiple channels. When using a VP connection, the VCI field can be used for this purpose, but the clients must negotiate the mapping of VCIs to logical channels. An example is shown in Figure 3, where clients **A** and **B** are using two logical channels. For the first, **A** sends on VCI = 3 and receives on VCI = 5. For the second, **A** sends on VCI = 5 and receives on VCI = 7. The VCIs used for each logical channel can be negotiated whenever a new channel is created. One method

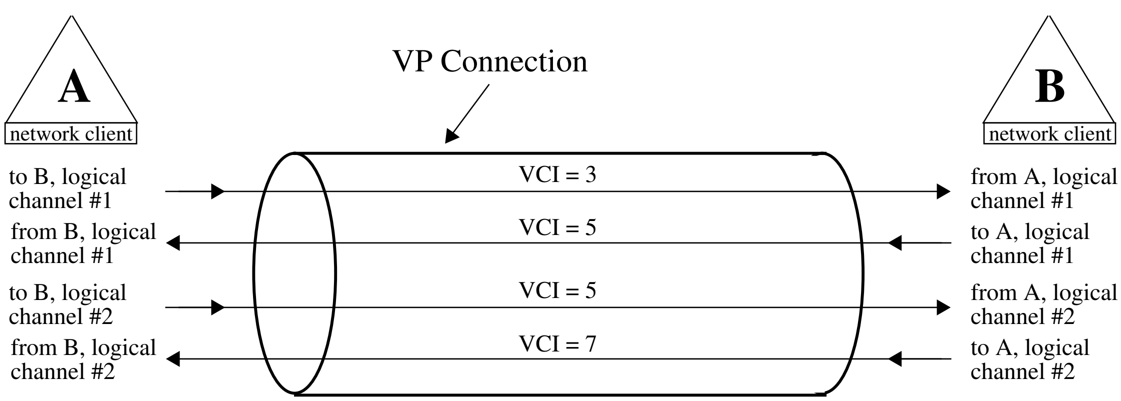


Figure 3. Use of Embedded Identifiers for Two Point-to-Point Logical Channels.

of negotiation is for the client initiating the operation to send the VCI that he wishes to receive on for this channel to the remote client (this message can be transmitted over a signaling channel established between the two, for example, using VCI = 0). The initiating client can easily select this VCI since he has complete knowledge of all arriving traffic and knows which VCIs are free. When the remote client receives the logical channel creation request, he chooses a VCI he wishes to receive on for the new channel and sends this back to the first client in an acknowledgment. The first client then sends a confirmation to the second, informing him that he now knows which VCI to use. From this point on, the clients can communicate over the new logical channel using the new VCIs just exchanged.

2.2.2 Multipoint Connections

If more than two clients are communicating on a connection, both the logical channels and the identity of each transmitter must be determined. As with point-to-point connections, the VCI can be used for this purpose, but the negotiation becomes more complicated. First, we consider the case of multiple clients communicating with a single logical channel, then discuss use of multiple logical channels.

One Logical Channel When multiple clients communicate over a single VP connection, the VCI can be used to differentiate between them, provided that each client sends on a unique VCI. An example is shown in Figure 4, where clients **A**, **B**, and **C** are communicating. Client **A** sends on VCI = 3, **B** sends on VCI = 5, and **C** sends on VCI = 9. When cells are received, the source is easily determined by the VCI. A negotiation is required when new clients join the connection so that each chooses a unique VCI. One option is for each new client to randomly choose an identifier, then broadcast the choice on the connection. If another client is already using this identifier or is attempting to allocate it to himself, he sends a negative acknowledgment, in which case the new client randomly chooses another identifier. The new client delays for a timeout period until all participants are likely to have responded. After the timeout period expires without a negative acknowledgment, the new client claims the new identifier as his own and broadcasts a message to all other clients informing them of the mapping. The choice of timeout period can be based on the expected network and processing latencies so that the resulting probability of incorrect VCI selection is acceptably low. Recovery procedures can be also added for when two clients choose the same VCI. Note that this scheme for selecting identifiers using timeouts is very similar to that used in the V system for creating new groups [5].

The problem of which VCI to use for the initial negotiation can be solved by establishing a well known signaling VCI. In this way, the negotiations do not conflict with other traffic. On the signaling VCI, per frame identifiers can be used to distinguish transmitters on this logical channel, as discussed in Section 2.1. The bandwidth used for connection joining is presumably low, making this choice feasible.

Multiple Logical Channels In the example above (Figure 4), the three clients may want to communicate using a second logical channel. This can be done by assigning a new set of VCIs for the second channel. For example, for the second logical channel, **A** could send on VCI = 13, **B** on VCI = 15, and **C**

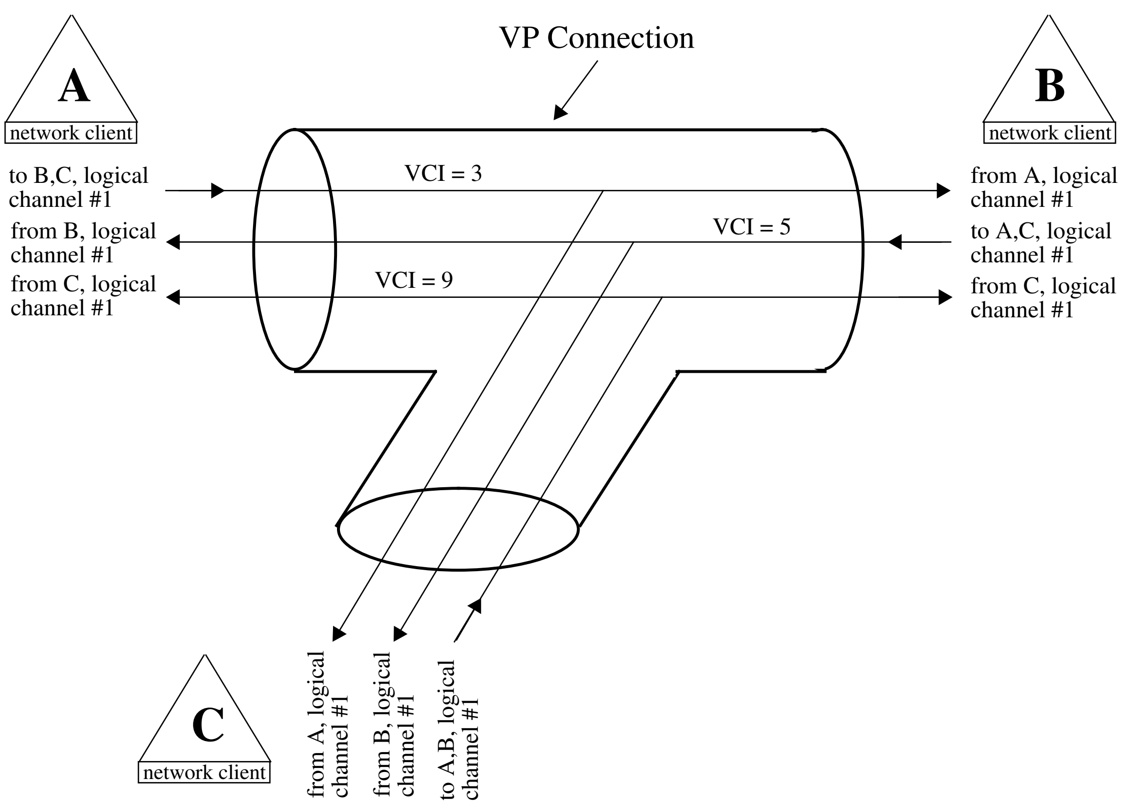


Figure 4. Use of Embedded Identifiers for a Single Multipoint Logical Channel.

on VCI = 19. On reception, each node can easily distinguish both the source of the transmission and the logical channel that it is associated with from the VCI. To create a new logical channel, one client could allocate an initial VCI that he will use when sending cells on this channel. This can be done using broadcast and timeout, as when joining the base connection. Other clients likewise join the new channel by broadcast and timeout, as with the base channel. In this way, new logical channels can form within the VP connection that only connect a subset of the clients. Since all cells are routed to all clients, the clients not participating in a given logical channel simply discard all cells associated with that channel (which can be done in the host interface hardware, Section 4).

2.3 Method 3: Use of Multiple Connections

The third method in which clients can multiplex and demultiplex multiple logical channels and multiple transmitters is to create separate ATM network connections for differentiation. With this approach, the cell VPI and/or VCI can be used to demultiplex cells that arrive intermixed on the link. This is identical to the demultiplexing technique of Section 2.2 (when VP connections are used), but differs overall in that we now have multiple ATM network connections. This solution might be desirable in circumstances where the network does not support VP connections and the client wants to avoid the overhead of embedded per cell identifiers. The primary drawbacks of this strategy are the network overhead of creating multiple connections and a potential difficultly in sharing bandwidth between the separate connections since many ATM networks allocate and enforce bandwidth on a per connection basis. This may lead to higher bandwidth usage and a higher probability of connection blocking [11]. More is said about this point in Section 3.

2.3.1 Point-to-point Connections

When two clients are communicating, a separate ATM network connection can be used for each logical channel. For example, Figure 5 shows clients **A** and **B** communicating using two logical channels, each in its own ATM network connection. With this configuration, the VPI and/or VCI that cells arrive on distinguishes the logical channel to which the cells are associated. To create a new logical channel, a client sends a signaling message into the network. The network then checks available resources and, if enough capacity exists in the network to support the new connection, it is created.

2.3.2 Multipoint Connections

When N clients (N > 2) are communicating, separate connections can be used to distinguish the clients by creating N one-to-(N–1) multicast unidirectional connections for each logical channel, one from each client to all others. For example, Figure 6 shows three clients communicating in this fashion. As above, the VPI and/or VCI that cells arrive on distinguishes the client that transmitted the cells. A problem arises when a new client is to be added. The N existing connections must be modified so that the new client is added as a new endpoint of these connections, and one new connection must be created from the new client to all other clients. For large multicast groups, the overhead of doing this may become prohibitive. However, for small multicast communication patterns (especially those which do not change over time), the overhead likely would be tolerable.

Creation of a new logical channel when using multiple connections is likewise complicated. If M of the N clients wish to participate in the new logical channel, M new one-to-(M-1) connections must be created (similar to the configuration in Figure 6) for the new logical channel. Once again, considerable network overhead is incurred in connection creation.

2.4 Use of Hybrid Configurations

The complexity of supporting multiple logical channels can be reduced by using a hybrid of the per frame identifier technique of Section 2.1 with either the per cell identifier technique of Section 2.2 or the separate connection technique of Section 2.3. Rather than using one of the base mechanisms of Section 2.2 or 2.3 for all logical channels, the first logical channel can be established in this manner, where the VPI and/or VCI differentiates the sender of each cell. Then, on this logical channel, additional

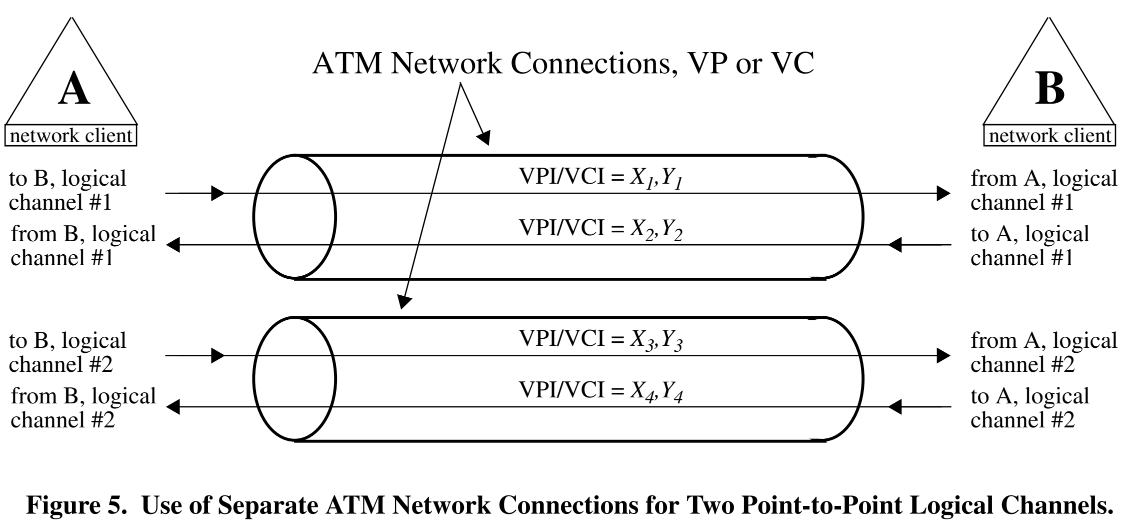


Figure 5. Use of Separate ATM Network Connections for Two Point-to-Point Logical Channels.

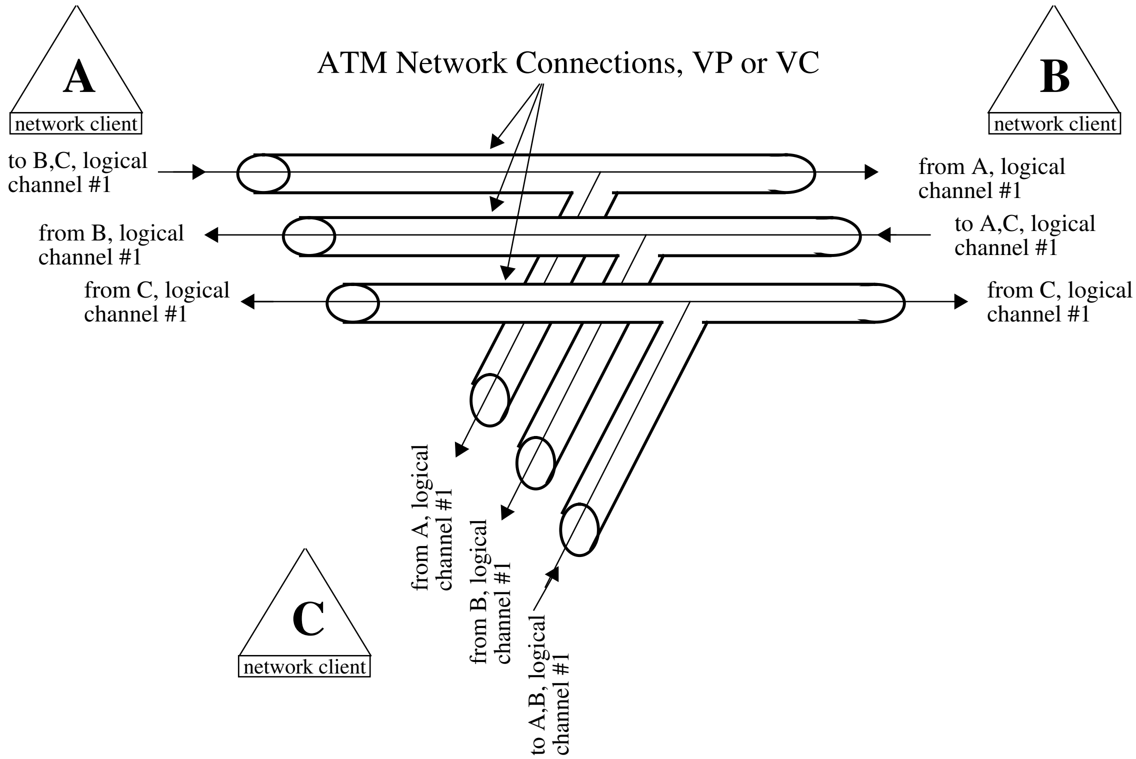


Figure 6. Use of Separate ATM Network Connections for a Multipoint Logical Channel.

logical channels can be layered using per frame identifiers, where, for example, the TCP port could be used to identify the logical channel to which each reassembled frame is associated. This hybrid approach requires that senders not intermix cells from different logical channels, and allows all cells to be reassembled if this requirement is met.

A second hybrid approach [8] uses per cell identifiers for determining the source of transmissions and a randomly chosen per cell identifier to differentiate cell streams. For example, if clients are communicating over a VP connection, whenever a client has a frame to send, it randomly chooses a VCI and sends all cells of this frame out on the VCI. Cells from different clients can intermix with this scheme and still be reassembled, provided the VCIs for each frame are all different. Only when two (or more) clients randomly choose the same VCI will reassembly be impossible for interleaved cells. The primary advantage of this hybrid scheme over using only per frame identifiers is that it reduces the probability of incorrect reassembly. One drawback of this scheme is that cells with all VCIs must be processed, as opposed to only processing cells on a small set of recognized VCIs (such as for only those logical channels in which the client is participating).

A third hybrid is a generalized grouping of the methods of Sections 2.1–2.3 and, possibly, the two hybrids mentioned above. For example, one client may use the per frame identifier technique of Section 2.1 for a low bandwidth name service application, the per cell identifier technique of Section 2.2 for a distributed talk application, and the separate connection technique of Section 2.3 for a video conference call application. Likewise, a single application may combine the various techniques, as appropriate, if it employs different communication paradigms.

3 Bandwidth Sharing

It appears that many ATM networks will have some bandwidth or quality of service reservation requirement, where clients request a particular amount of bandwidth or a particular service class when creating connections [11]. The network then uses this information to determine whether the client's request can be supported with the available resources, and enforces resource usage so that the clients do not exceed their requested usage.

When clients communicate with multiple logical channels and when more than two clients communicate over one or more connections, it is often desirable that the logical channels and connections all share a common bandwidth pool. Two reasons for this are: 1) only a single aggregate bandwidth specification need be given for the total communication requirement, rather than one for each logical channel or separate connection, and 2) multiple clients who communicate in high bandwidth bursts require a lower per capita bandwidth and can transmit at higher peak rates.

To see where bandwidth sharing is advantageous, consider an example where N client workstations connect to a file server. Ideally, we would like all clients to be able to share the total capacity of the network (or some fraction thereof) in such a manner that each client can transfer at or near capacity when it has data to send. One way in which this can be accomplished is to set up a single (N+1)-way VP connection between all of the workstations and the server. Within this connection, we create N bidirectional point-to-point logical channels, one between each client and the server. The VCIs used for each logical channel are different, thereby allowing the server to distinguish from whom each request was sent and allowing the clients to discard responses directed to other clients (something that is easily supported in the host interface hardware, Section 4). Since only a single ATM network connection is used, bandwidth sharing is implicit. By setting the bandwidth requirements to the link capacity when the connection is created, the clients can effectively use the full capacity of the network.

An alternative way in which the client/server paradigm could be supported is by creating separate point-to-point ATM network connections between each client and the server. As above, the VPI and/or VCI can be used by the server to distinguish which client sent the request. Client interfaces do not need to discard responses for other clients since these responses will only propagate to the intended client. However, bandwidth sharing is only possible if the network allows connections to be grouped for this purpose. It is not clear whether any network will provide this type of grouping since it complicates both the network hardware and software. If we assume this type of grouping is not possible, then the clients may not be able to achieve the same level of performance from the network as if they shared a single connection. The reason is that the bandwidth on the link running from the network to the server cannot exceed the link's capacity. In the worst case, each of the N clients would be limited to a bandwidth of L/N, where L is the link's capacity. In practice, a higher allocation for each client would probably be possible due to the statistical nature of the connections, but this level would likely be much less than that available if all clients share the same bandwidth pool. An additional drawback of using multiple connections is that each client must specify its bandwidth requirement when creating the connection to the server. Without knowing how many other clients are connected to the server, it is difficult for the client to know what bandwidth to specify. Also, as the number of clients changes, the clients would ideally need to alter their bandwidth requirement to make optimal use of the network so that bandwidth is available for new clients to connect to the server. This is something that is not required with the single VP connection approach.

4 Host Interface

We describe a host interface architecture capable of supporting both VP and VC connections. The principle of operation is to decode incoming cells based on only the VCI, while filling in both the VCI and VPI in outgoing cells. So long as the same VCI is not used for arriving cells in two separate VPI connections, this technique allows all cells to be associated with the correct connection. Additionally, since only the VCI is used for decoding, the lookup table can be kept small (64K maximum entries) and the lookup operation is simpler than if both the VPI and VCI were used.

A block diagram of the host interface is shown in Figure 7. This interface performs segmentation and reassembly of frames using an *end* indication embedded in the header [9] (Section 2). ATM cells enter the interface at the top left. After being converted to electrical signals and synchronized to the internal clock speed, the cells enter a header decoder. Here, the VCI is read from the cell and a table lookup operation is performed based on the VCI. (We assume that the host memory is used for the lookup table, although it is also possible to use a dedicated memory on the host interface.) For each VCI, the interface maintains:

1) An indication of whether the VCI is currently allocated (in use).
2) The *state* of an interface *reassembly state machine*: *idle* or *active*.
3) The partial CRC computed on cells received thus far, for the frame being reassembled.
4) The memory address for storage of the next byte of the frame.
5) The memory address of the end of the reassembly buffer.
6) A timer indicating when the most recent cell for this frame was received [3].

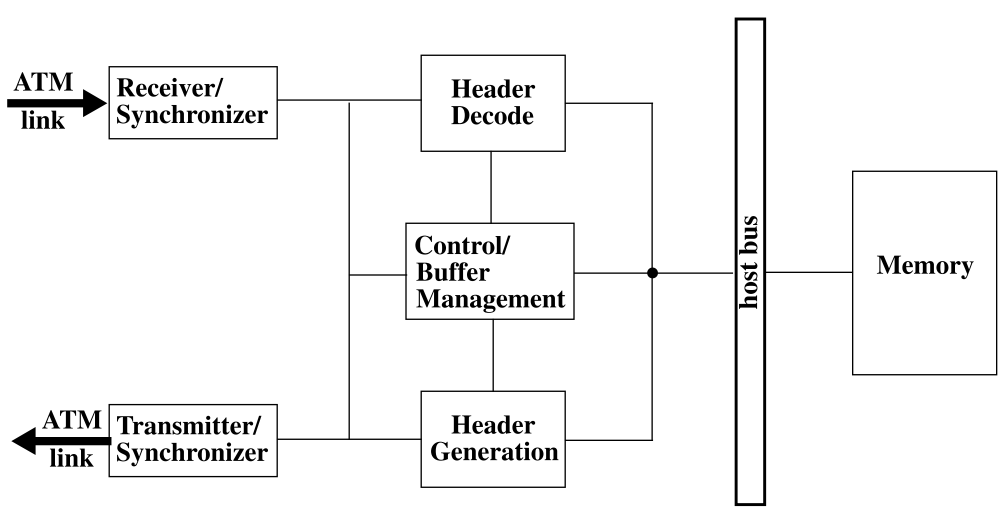


Figure 7. ATM Host Interface.

The interface discards cells that arrive on unallocated VCIs. If a non-*end* cell is received on an allocated VCI while in the *idle* reassembly state, a transition is made to the *active* state. These and other cells that arrive while in the *active* state are placed in the memory (minus the header) at the address pointed to by the buffer pointer. For each non-*end* cell received the partial CRC is updated, the location for storage of the next byte is advanced, and the VCI's timer is reset. After an *end* cell is processed or when the reassembly buffer fills, the host is notified of the arrival, its location, an indication of whether the complete frame was received or that the buffer filled before the *end* cell was received, and an indication of whether the CRC was valid (for complete frames). An *end* cell causes a transition to the *idle* reassembly state, but a full buffer does not, as the remainder of the frame is placed in the next receive buffer (if present). If the VCI's timer expires, the reassembly state machine returns to the *idle* state and the buffer is reused for another reassembly. If an *end* cell is received while in the *idle* state, it is treated as a single cell frame and delivered to the host, without entering the *active* state.

The *control/buffer management* block reads commands that the host has placed in the memory. These commands are used to: request transmission of frames, update table entries, and supply the interface with reassembly buffers. When frames are to be transmitted, the host includes the starting address of the data to be transmitted, the amount of data to be transmitted, and the VPI/VCI values to be placed in the outgoing cells. After the interface completes the transmission, it signals the host so that the buffer may be reused. When the host wishes to begin receiving frames on a new VCI, it issues a table update command that instructs the interface to allocate the specified VCI. Reassembly buffers are passed to the interface from the host. The header decoder requests buffers from the buffer manager whenever the first cell of a new frame arrives and whenever the buffer of a frame being actively reassembled fills up.

Note that the description of the host interface given here is by no means complete. We have omitted many details for the sake of brevity. Also, we have described a simple interface that could be enhanced to provide increased functionality. One final note, a very similar host interface to the one described here could be built using an embedded identifier, such as the type 4 AAL MID, for cell stream demultiplexing and reassembly.

5 Summary

The ATM standard provides for two types of connections: Virtual Path (VP) and Virtual Channel (VC). These connection types can be used in a variety of ways to support one-to-one, one-to-many, many-to-one, and many-to-many communication between network clients. While either one of these connection types is sufficient to support all of these communication paradigms, some paradigms are better suited for VP connections and others are better suited for VC connections. Consequently, we believe that network providers should extend both connection types to clients, and host interfaces should be designed to support both types.

Acknowledgments

The idea of using per frame, higher level protocol information for source discrimination on a multicast connection (without per cell identifiers) was first presented to us by Brian Lyles (Section 2.1). He also suggested the use of randomly chosen VCIs to reduce the probability of conflict (Section 2.4).

References

[1] ANSI T1S1 Technical Sub-Committee. *Broadband Aspects of ISDN Baseline Document*. T1S1.5/90-001, June 1990.

[2] R.G. Bubenik, J.D. DeHart, and M.E. Gaddis. "Multipoint Connection Management in High Speed Networks." In *IEEE Infocom '91: Proceedings of the Tenth Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 59-68, April 1991.

[3] R.G. Bubenik, J.D. DeHart, and M.E. Gaddis, *A Strategy for Layering IP over ATM*, Washington University Applied Research Laboratory Working Note 91-04-Version 1.0, April 5, 1991.

[4] CCITT. *Recommendations Drafted by Working Party XVIII/8 (General B-ISDN Aspects) to be Approved in 1990,* Study Group XVIII—Report R 34, June 1990.

[5] D.R. Cheriton and W. Zwaenepoel, "Distributed Process Groups in the V Kernel." *ACM Transactions on Computer Systems*, 3(2):77-107, May 1985.

[6] M.E. Gaddis, R.G. Bubenik, and J.D. DeHart. "Connection Management for a Prototype Fast Packet ATM B-ISDN Network." In *Proceedings of the National Communications Forum*, pages 601-608, October 1990.

[7] M.E. Gaddis, R.G. Bubenik, and J.D. DeHart. "A Call Model for Multipoint Communications in Switched Networks." Submitted for publication.

[8] B. Lyles, Xerox PARC. Private Communication, April 12, 1991.

[9] T. Lyon, "Simple and Efficient Adaptation Layer (SEAL)." Proposal T1S1.5/91-292 to ANSI working group T1S1.5, August 1991.

[10] S.E. Minzer. "Broadband ISDN and Asynchronous Transfer Mode (ATM)." In *IEEE Communications Magazine*, 27(9):17-24, September 1989.

[11] J.S. Turner. "A Proposed Bandwidth Management and Congestion Control Scheme for Multicast ATM Networks." Washington University, Computer and Communications Research Center, Technical Report-WUCCRC-91-01, May 1991.