February 1993

A Host Interface Architecture and Implementation for ATM Networks

C. Brendan S. Traw
University of Pennsylvania

Recommended Citation

C. Brendan S. Traw , "A Host Interface Architecture and Implementation for ATM Networks ", . February 1993.

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-93-30.

A Host Interface Architecture and Implementation for ATM Networks

Abstract

The advent of high speed networks has increased demands on processor architectures. These architectural demands are due to the increase in network bandwidth relative to the speeds of processor components. One important component for a high-performance system is the workstation-to-network "*host interface*". The solution presented in this thesis migrates a carefully selected set of protocol processing functions into hardware. The host interface is highly parallel and all per cell functions are performed by dedicated logic to maximize performance. There is a clean separation between the interface functions, such as segmentation and reassembly, and the interface/host communication. This architecture has been realized in a prototype which connects an IBM RISC System/6000 workstation to a SONET-based ATM network carrying data at the OC-3c[1] rate of 155 Mbps.

Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-93-30.

A Host Interface Architecture and Implementation For ATM Networks

MS-CIS-93-30
DISTRIBUTED SYSTEMS LAB 28

Chandler Brendan Stanton Traw

University of Pennsylvania
School of Engineering and Applied Science
Computer and Information Science Department
Philadelphia, PA 19104-6389

February 1993

UNIVERSITY OF PENNSYLVANIA
THE MOORE SCHOOL OF ELECTRICAL ENGINEERING
SCHOOL OF ENGINEERING AND APPLIED SCIENCE

A HOST INTERFACE ARCHITECTURE AND IMPLEMENTATION FOR ATM NETWORKS

Chandler Brendan Stanton Traw

Philadelphia, Pennsylvania

May 1992

A thesis presented to the Faculty of Engineering and Applied Science of the University of Pennsylvania in partial fulfillment of the requirements for the degree of Master of Science in Engineering for graduate work in Electrical Engineering.

David J. Farber
(Advisor)

Sohrab Rabii
(Graduate Group Chair)

Abstract

The advent of high speed networks has increased demands on processor architectures. These architectural demands are due to the increase in network bandwidth relative to the speeds of processor components. One important component for a high-performance system is the workstation-to-network *"host interface"*. The solution presented in this thesis migrates a carefully selected set of protocol processing functions into hardware. The host interface is highly parallel and all per cell functions are performed by dedicated logic to maximize performance. There is a clean separation between the interface functions, such as segmentation and reassembly, and the interface/host communication. This architecture has been realized in a prototype which connects an IBM RISC System/6000 workstation to a SONET-based ATM network carrying data at the OC-3c[1] rate of 155 Mbps.

[1] OC-n refers to multiples of a base rate of about 52 megabits per second: thus the OC-3c bandwidth is 155 Mbps.

Acknowledgements

I would like to acknowledge the contributions of the following people: David Farber and Jonathan Smith for excellent guidance as advisors for this work. I would also like to thank Jonathan Smith for employing his kernel hacking ability to construct device drivers for the Host Interface. Bruce Davie for exposing me to Host Interfacing and providing guidance on the architecture and its written presentation. Al Broscius and Sanjay Udani for stimulating discussions. Taso Devetzis and Mike Maszczak for loaning me "one more EPLD." And finally Brianna Nagle for supporting me through the long hours in the lab.

AURORA is a joint research effort undertaken by Bell Atlantic, Bellcore, IBM Research, MIT, MCI, NYNEX, and Penn. AURORA is sponsored as part of the NSF/DARPA Sponsored Gigabit Testbed Initiative through the Corporation for National Research Initiatives. NSF and DARPA provide funds to the University participants in AURORA. Bellcore is providing support to MIT and Penn through the DAWN project. IBM has supported this effort by providing RISC System/6000 workstations. The Hewlett-Packard Company has supported this effort through donations of laboratory test equipment.

Contents

List of Figures

1 Introduction

Processor speeds and workstation architectures have been improving rapidly, but not sufficiently quickly to keep up with the tremendous increases in network bandwidths which are becoming available to hosts. In particular, bandwidths of experimental network are approaching a billion bits per second. To assist hosts with the protocol processing and data movement tasks associated with these high bandwidth network connections, a new generation of host interface architectures is necessary.

1.1 Host Interfaces and Protocol Architectures

Protocol architectures can be viewed as a *stack* of *layers.* The ISO OSI model, for example, consists of seven layers [27]. When implemented, the protocol layers need not observe the separation of the logical model. The physical layer must consist of hardware by definition, but the implementor can make hardware versus software implementation decisions for each succeeding layer.

Software is often used when flexibility or tuning are required. As the behavior of a layer becomes better understood, functionality can be migrated from software to hardware. The benefit is twofold. First, protocol processing overhead is offloaded from the host. This frees the host to address applications workload, and provide concurrent processing. If the computers are high-performance workstations, and not supercomputers, this is a significant attraction. Second, specialized hardware can often perform functions faster than the host, thus increasing the bandwidth available to applications.

Typically, host interfaces for high speed network connections must be placed architecturally "close" to the processor memory bus to achieve high performance. This is due both to latency and to the delay imposed by multiple stages of processing. These stages can be implemented in hardware or software. We have consistently chosen to implement fixed decisions in the interface hardware and leave unmade decisions to host software.

1.2 AURORA

The host interface work at Penn has been centered on developing a high-performance host interface for workstation hosts in the AURORA Gigabit Testbed environment. [10] AURORA is an experimental wide area network testbed [26] whose main objective is the exploration and evaluation of network technologies. The Gbps network will link four sites:

- Bellcore's Morristown Research and Engineering Laboratory in Morristown, NJ
- IBM Research's Computer Science Laboratory in Hawthorne, NY
- MIT's Laboratory for Computer Science in Cambridge, MA
- University of Pennsylvania's Distributed Systems Laboratory in Philadelphia, PA

The logical topology of AURORA is illustrated in Figure 1.

As illustrated, only one location (IBM Research) has supercomputing capability, thus we will not be interconnecting supercomputers. Rather, the connections are from supercomputer to workstation, remote display to workstation, and workstation to workstation. Our focus on workstation technology is driven by three factors:

1. Supercomputers are expensive, and *networking* research questions rarely motivate their purchase.
2. Interesting networks are large in both bandwidth and scale. The rarity of supercomputers limits the ability of testing scalability with interconnected supercomputers.
3. Workstations will be the predominant processor class connected to such networks.

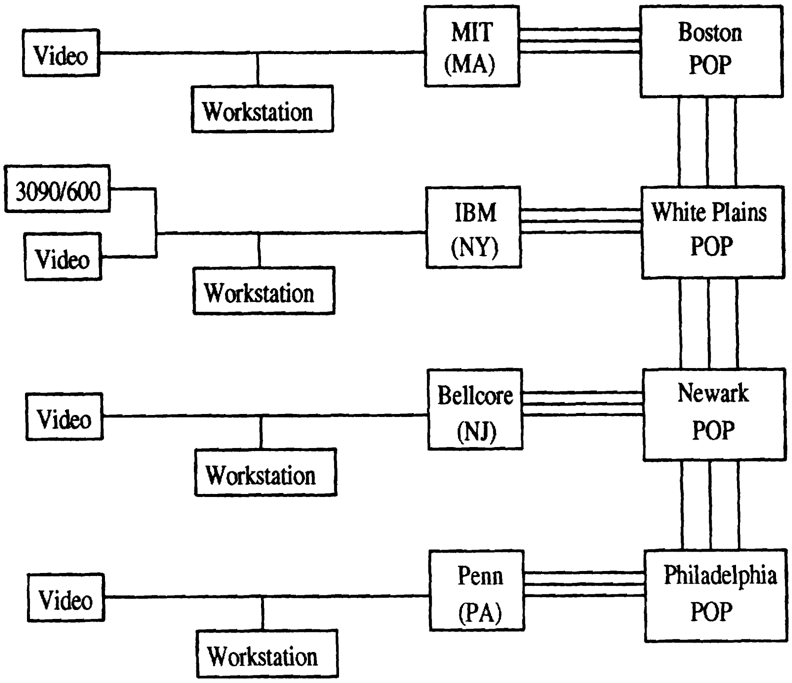


Figure 1: AURORA Logical Topology

Each of the three lines from sites to central offices in Figure 1 represents an OC-12. The point of this configuration is to first build independent plaNET (plaNET is the follow-on to PARIS [8]) and Sunshine [19] networks. These independent networks will later be interconnected in order to understand interoperability of the technologies. Our host interface [29] is intended for the Sunshine-ATM logical topology.

1.3 Goals and Design Philosophy

The research goals are as follows:

1. A hardware/software architecture which is flexible and allows experimentation with portions of the protocol stack.
2. A focus on architectural solutions to achieve good cost/performance, so our results scale across technology choices.
3. Low absolute cost, so that large-scale replication is possible.

The resulting host interface should meet all three goals. The design philosophy for our architecture is based on providing a "common denominator" set of services in dedicated hardware. All per cell activities such as CRC creation and verification, segmentation, and reassembly are performed by the host interface in hardware. The host is responsible for all higher level activities.

By only supporting a subset of the services which may be required for a particular application, the host interface is not capable of completely relieving the host of all protocol operations. We feel that this is reasonable since we are able to maintain flexibility in protocol implementation in exchange for some overhead incurred by host software. Protocol flexibility is important for the following reasons:

- Not all protocols and applications are defined yet.
- Services are extremely varied. It would be difficult, for example, to provide support for all possible adaptation layers in dedicated hardware.

2 Related Work

Several research projects are targeted towards high-performance host interfaces. One major difference between these implementations is the number of protocol processing functions which the host interface performs.

One important focus has been interfaces which accelerate transport protocol processing [32]. For example, Kanakia and Cheriton's [23] VMP Network Adapter Board serves as a hardware implementation of Cheriton's Versatile Message Transaction Protocol (VMTP). Abu-Amara, *et al*, [3] can target any set of protocol layers (to the degree that they can be precisely specified) with their PSi silicon compiler approach. With this method, the protocol is specified using a symbolic programming language, and mask descriptions for VLSI fabrication are generated as output of the compiler. The Nectar Communications Accelerator Board (CAB) [5] can be programmed with various protocols.

The CAB communicates with the host memory directly, and the programmability can conceivably be used by applications to customize protocol processing. Cooper, *et al* [12] report that TCP/IP and a number of Nectar-specific protocols have been implemented on the CAB connected to Sun-4 processors.

Another approach has been explored for ATM interfaces, which puts minimal functionality in interface hardware [11]. This approach is characterized by the assignment of almost all tasks to the workstation host including adaptation layer processing. It has two potential failings. First, RISC workstations are optimized for data processing, not data movement, and hence the host must devote significant resources to manage high-rate data movement. Second, the operating system overhead of such an approach can be substantial without hardware assistance for object aggregation and event management. Somewhat more functionality is achieved in the interface designed for the Cambridge Backbone Ring. [20] Such approaches can take significant advantage of aggressive workstation technology improvements.

Penn's Micro Channel Architecture host interface is not the only one being designed for the AURORA Testbed environment. Davie of Bellcore [15] reports on a host interface design for the TurboChannel bus of the DECStation 5000 workstation [16]. The design relies on Intel 80960 RISC microcontrollers to perform the protocol processing and flow control for a trunk group of four STS-3c lines (622 Mbps). Powerful offboard engines are attractive from a parallel processing point of view, since they migrate processing and data movement control away from the host CPU. In addition to this performance, significant flexibility is gained from the reprogrammability of the host interface behavior. However, this approach is costly, and extremely careful programming is required to achieve tight performance goals, especially when portions of multiple protocol stacks must be supported.

The Penn implementation provides much of the same functionality as the interface being developed by Davie (from which many of our ideas about cell management functions are derived), while exploring an alternative host interface hardware/software organization.

3 Host Interface Architecture and Implementation

3.1 Implementation Technology Choices

As we focused on architectural issues, we have chosen to implement this architecture with relatively low cost, commercially available logic and memories. To minimize space and power requirements, high density, erasable, programmable logic devices are used [2]. By avoiding high clock speeds and more exotic technologies such as emitter coupled logic (ECL) or custom VLSI, the emphasis can be kept on architectural rather than technological choices. Re-implementations of this architecture using such technologies should result in significantly higher performance than reported here. The one major exception to this policy is the use of a custom VLSI SONET framer [25] which was developed at Bellcore.

The resulting implementation without the optoelectronic components necessary to interface to the OC-3c network connection consists of two wire-wrapped 32 Bit Micro Channel cards, shown below Figure 2. The segmenter and SONET interface occupy one card while the reassembler occupies the other. Power consumption of the host interface (without optics) is less than 35 watts.

3.2 Networking Environment

To provide good support for connectionless traffic on the network, we have chosen to provide hardware support for the Class 4 ATM Adaptation Layer [22] (AAL4). The use of other adaptation layers is not prohibited by this extra support for the AAL4. The extra processing required for the support of other adaptation layers will have to be borne by the host processor.

Several assumptions are made about the environment in which this host interface will be operating. First, we assume that the the network will experience very low cell loss and corruption rates, thus error recovery will be a rare event. Such rare events can be

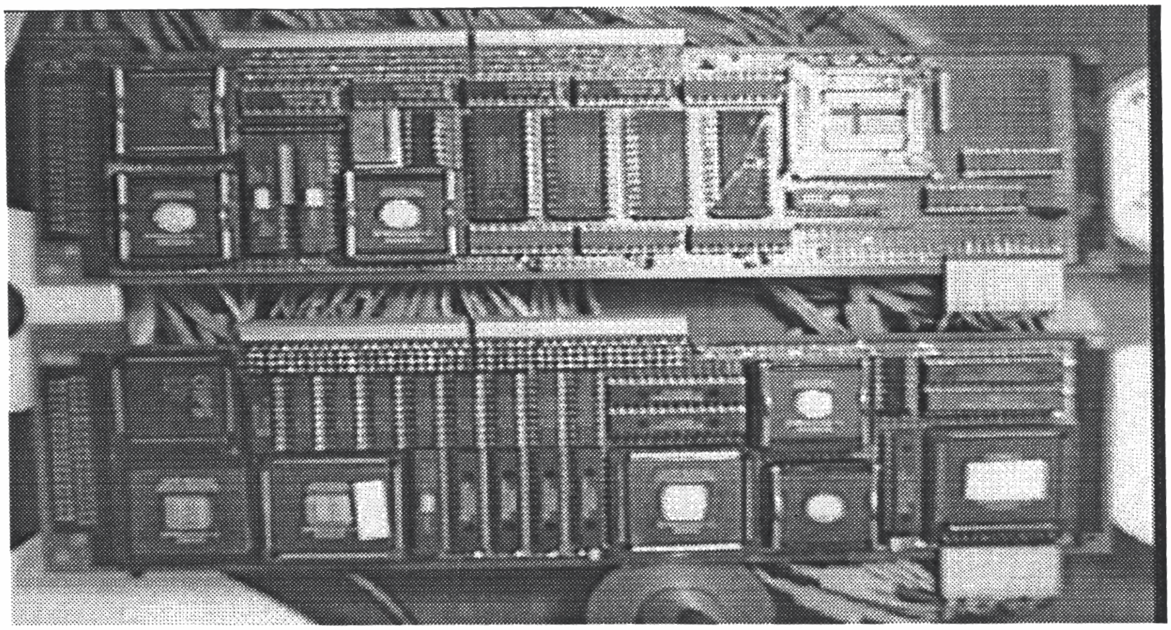
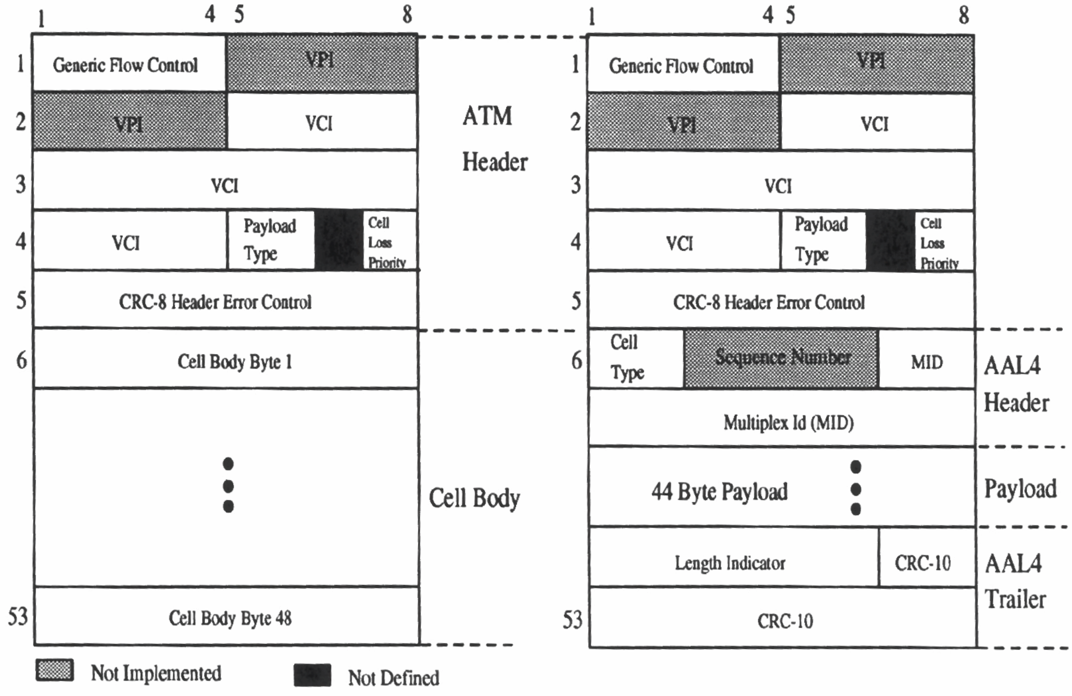


Figure 2: Picture of Reassembler (Bottom) and Segmenter (Top)

costly operations to perform. We also assume that there will be no cell misordering in the network. These two assumptions allow us to ignore the AAL4 segment number to the receive side of the interface. For compatibility with other equipment, AAL4 segment numbers will be generated by the segmenter. Cell loss for connectionless data using the AAL4 would be detected at the AAL4 Convergence Sublevel (CS) by a mismatch between the actual length of the CS-PDU and the CS-PDU's length field. The Virtual Path Identifier (VPI) portion of the ATM header [21] is also ignored in the AURORA ATM environment, thus it is not supported in the initial prototype of the host interface. Finally, the most significant bit of the Virtual Circuit Identifier (VCI) in the ATM header is used to indicate that a particular virtual connection is transporting AAL4 data. Figure 3 illustrates the ATM cell formats used.



ATM Format

ATM Adaptation Layer 4 Cell Format

Figure 3: Cell Formats

3.3 Micro Channel Architecture Bus

The Micro Channel Architecture Bus [13] on the RISC System/6000 [6] has been chosen as the host interface's point of interface for the following two reasons. First, it provides a relatively high bandwidth data path into the host's main memory and to other peripherals on the workstation such as video capture cards. Secondly, the Micro Channel Architecture bus is non-proprietary and relatively easy to connect to in comparison to the RISC System/6000's memory bus.

Commands and status are exchanged between the host interface and the host CPU by standard I/O write and read bus transfer cycles. The host interface is capable of acting as a 32 bit streaming bus master. Streaming is a modified bus transfer cycle, which begins as a standard bus cycle, but allows contiguous words in the address space to be transferred every 100 ns (320 Mbps peak bandwidth) once the transfer has been started.

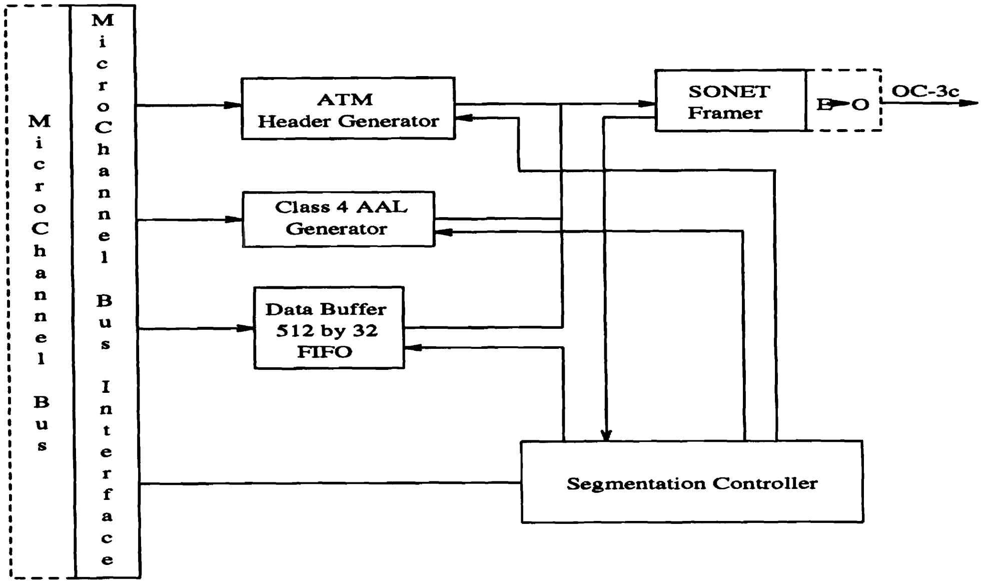


Figure 4: Segmenter

Thus, the time required to initiate the transfer can be amortized over many word transfers. Being a bus master allows the host interface to transfer data to and from the host's main memory independently of the host CPU. It also allows data to be transferred directly between other peripherals and the host interface without host CPU intervention.

The Micro Channel Architecture Interfaces used for the Segmenter and Reassembler are very similar. Both are based on the Chips and Technologies 82C612 DMA Slave Controller [7]. Additional logic has been added to this controller to make it capable of being a bus master for 32 bit streaming transactions.

3.4 The Segmenter

A block diagram of the Segmenter is illustrated in Figure 4. The Segmenter provides the capability to read data from the host's main memory (or other data source located on the Micro Channel Bus such as a video capture peripheral card), segment it into ATM cells, and then transmit it into the network at the OC-3c data rate of 155 Mbps.

When data is to be transmitted, the host must first load several control registers with data: the source address, length, and ATM header control fields to be used, such as the VCI. If the VCI indicates that the data is to be transmitted using the AAL4, the MID must also be specified.

Once this information is available, the segmenter concurrently generates the ATM header CRC and initiates the streaming data transfer from the source of the data across the Micro Channel bus to the Segmenter. As soon as sufficient data has been transferred into the Segmenter's data buffer, one cells worth of data is extracted from the buffer by the Segmentation Controller and is concatenated with an ATM header. An AAL4 header and AAL4 trailer are also added if appropriate. Both the CRC-8 (for the ATM header) and the CRC-10 (for the AAL4 trailer) are calculated at a rate of a byte per clock cycle as the cell header and body are passed to the SONET framer. This process is repeated until the entire block of data has been transmitted.

3.5 Reassembler

The Reassembler is presented in Figure 5. The Reassembler is able to receive data from the OC-3c network connection, reassemble it, and then deliver the reassembled data to the host's main memory or to another peripheral card on the Micro Channel bus.

To read data reassembled by the host interface, the host must specify to the destination of the data, the internal list reference number of the connection/datagram, and the number of cells to be transferred. The origin of the internal list reference will be discussed shortly in the CAM Lookup Controller section.

The Reassembler is composed of five major functional units which all work concurrently. Four of the units, the Cell Manager, CAM Lookup Controller, Linked List Manager, and Dual Port Reassembly Buffer Controller form an ATM cell-processing "pipeline." Only control information is passed through this pipeline inorder to minimize the buffer space required in the pipeline and to avoid repetively copying the cell body data from stage to stage.

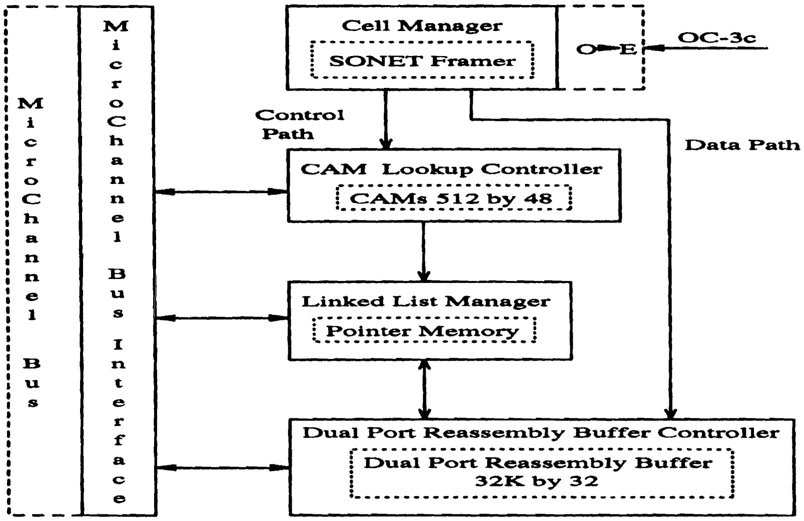


Figure 5: Reassembler

3.5.1 Cell Manager

The Cell Manager verifies the integrity of the header and payload (if the call is carrying AAL4 data) of the cells that are received by the SONET framer interface to the network by calculating the CRC-8 of the ATM header and CRC-10 of the ATM cell body and comparing them with the values in the cell just received. If the values match, the cell is assumed to be intact. The Cell Manager then extracts the VCI from the ATM header and the MID, segment type, and length indicator from the AAL4 header and trailer. While these fields are being extracted and the CRCs are being verified, the cell body is placed in a FIFO buffer for later movement into the dual port reassembly buffer. Since the cell body will be placed into the FIFO buffer before its integrity can be verified, the Cell Manager can request that the body be flushed from the FIFO by the Dual Port Reassembly Buffer Controller. These operations take exactly one celltime, 2.7 μs at the OC-3c rates.

3.5.2 CAM Lookup Controller

The CAM Lookup Controller (CLC) manages two 256 entry (48 bits) content addressable memory (CAM) devices from AMD[1]. One is reserved for virtual circuit traffic while the other is reserved for datagrams. Thus, 256 virtual circuit connections and 256 datagrams can be reassembled simultaneously. Virtual circuits are identified by their VCI while datagrams are identified by their VCI and MID. We considered using direct lookup RAM tables instead of CAMs but decided against this option since for datagrams, the address space is 26 bits (16 bit for VCI + 10 bits for MID). Larger CAM are available if the 256 connection/datagram limit proves to be confining.

When a VCI or VCI+MID is received from the Cell Manager, the CLC searches the appropriate CAM for a matching entry. If none is found and an unused entry is available, the CLC assumes that the identifiers belong to a newly established connection or datagram and writes the identifiers into the empty location. If no entry is available, the cell is dropped. Provided that a match was found, or a new entry was created, the CLC passes the location of the match or new entry to the Linked List Manager. This location is used as the internal list reference number for the connection or datagram.

The host is able to read the contents of each CAM entry to associate internal reference numbers with their corresponding VCI or VCI+MID. The host is also able to delete entries which are no longer active.

The CLC requires a maximum of eleven 50 ns clock cycles (550 ns) to perform the processing required for a cell.

3.5.3 Linked List Manager

The Linked List Manager (LLM) constructs and updates the linked list datastructures responsible for reassembly. These data structures are stored in a 32K by 16 static RAM.

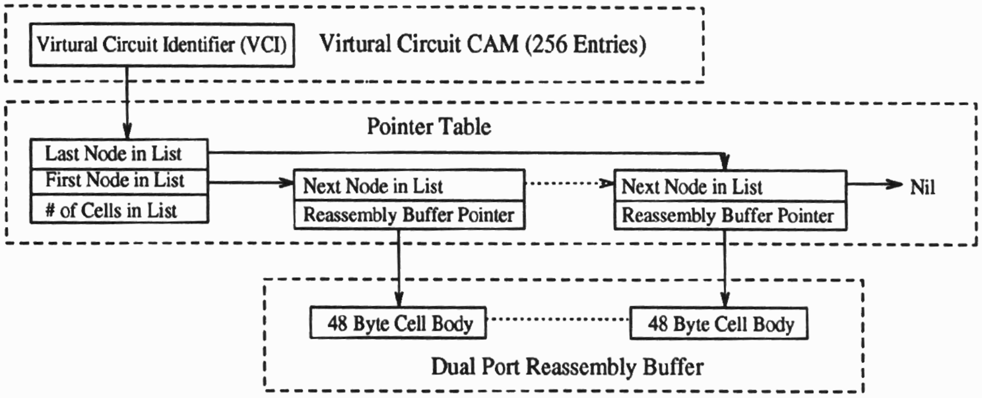

Virtural Circuit Identifier (VCI)
Virtural Circuit CAM (256 Entries)
Pointer Table
Last Node in List
First Node in List
# of Cells in List
Next Node in List
Reassembly Buffer Pointer
Next Node in List
Reassembly Buffer Pointer
Nil
48 Byte Cell Body
48 Byte Cell Body
Dual Port Reassembly Buffer

Figure 6: Control Structures for Virtual Circuit Reassembly

We believe that linked lists are an excellent mechanism for performing reassembly for two reasons. First, they allow dynamic allocation of memory. Extremely active connections can allocate more memory than their less active counterparts. Secondly, since each linked list node has a cell body sized portion of the dual port reassembly buffer associated with it, all manipulations of the dual port reassembly buffer are controlled by the linked list datastructures. Thus, the data stored for a connection or datagram can appear contiguous without being physically contiguous in the reassembly buffer.

The LLM is capable of performing the following functions on the linked lists:

- Delete a list
- Append a node to the end of a list
- Remove a node from the front of a list

Each of these operations also updates the list status information at the head of the list affected.

During configuration, the host is able to read and write into the RAM containing the datastructures. This capability is necessary to initialize the data structures prior to host interface operation. During operation, the host is only able to read the status blocks

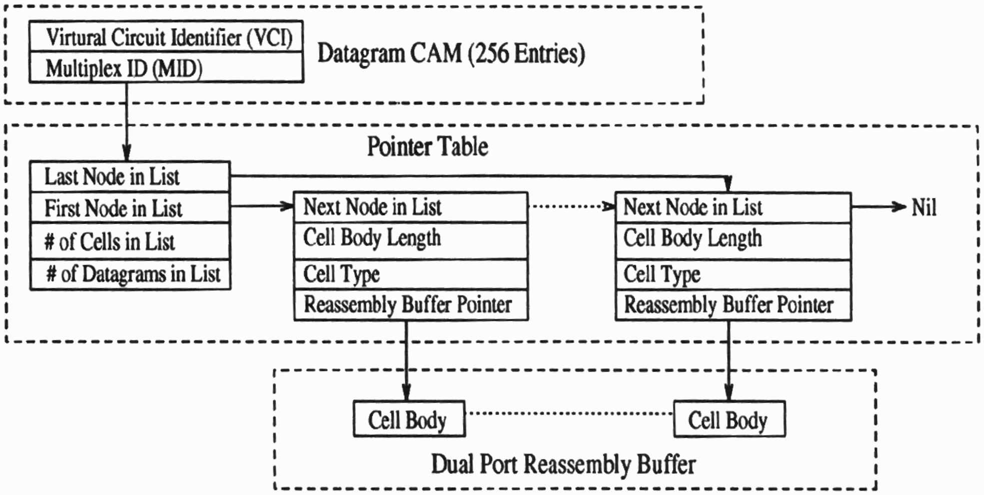


Figure 7: Control Structures for Datagram Reassembly

at the beginning of each list to remain aware of the network activity. The LLM is responsible for all manipulation of the lists during operation.

When the internal list reference number is passed to the LLM from the CLC, the LLM appends a new node at the end of the list specified. The pointer to the portion of the dual port reassembly buffer assigned to the node just appended to the list is passed to the dual port reassembly buffer controller.

When the host reads data from the host interface, nodes are removed from the front of the affected list, and the reassembly buffer pointers are passed to the dual port reassembly buffer controller so that the appropriate data can be move from the host interface.

The host, through the CLC, is able to request that a particular list be deleted.

In the worst case, the LLM requires thirteen 50 ns cycles (650 ns) to process a cell.

3.5.4 Dual Port Reassembly Buffer Controller

The Dual Port Reassembly Buffer Controller (DPRBC) is the final stage of the ATM cell processing pipeline. It is responsible for moving data to and from the dual port reassembly buffer. This buffer consists of a single ported 32K by 32 RAM bank which is effectively dual ported by the DPRBC. Dual port RAMs are commercially available, but they are less dense and more expensive than the single port RAMs used.

The DPRBC is able to move a cell body from the FIFO associated with the Cell Manager into the reassembly buffer in 2.4 μs (cell time is 2.7 μs). A cell body can be extracted from the buffer for movement across the bus in 1.2 μs, the minimum time required to move the data across the bus.

4 Host Software Support

A number of assumptions have been made about the host software, particularly the host operating system's *active* management of the host interface. Active management is assumed due to the following observations:

1. Unlike mainframes, supercomputers or minicomputers, workstations are rarely a shared resource

2. Egalitarian scheduling policies have made real-time awkward

3. Interrupt-handling overhead is large (for example, a save/restore of the RISC System/6000's registers is 256 bytes versus the 48 byte ATM payload) and effects a significant reduction in cache [24] effectiveness. Full interrupt service per ATM cell would severely limit the workstation's network bandwidth.

4. The general solution to this problem is to use more aggressive I/O device management policies and scheduling strategies. An example would using an

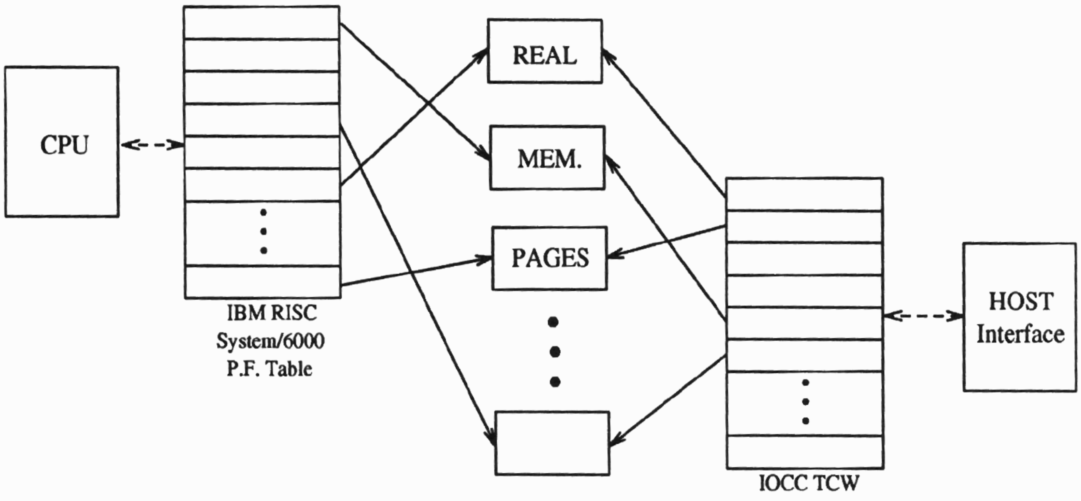


Figure 8: Illustration of TCW and PFT Usage

interrupt only as an event indicator. Actual transfer of bursts[2] of ATM cells would be accomplished in a scheduled manner.

The current host interface support software [30] consists of AIX character-special [28] device drivers.

The driver can be configured into the system at boot time if the device is detected, or later under program control. The host interface presents a unique device identifier when probed, and this identifier is used to gather descriptive information (including driver routines) from a system object database. Configuration includes allocating addresses for use by the device; the device uses these addresses for its control registers and to support streaming mode transfers.

The software prepares for streaming by initializing a number of translation control words (TCWs) [14] in the Micro Channel's I/O Channel Controller (IOCC). In addition, page mappings are adjusted for pages in the host memory; the RISC System/6000 uses an Inverted Page Table also referred to as the Page Frame Table (PFT). The TCWs and Page Frame Table entries allow both the device and the CPU to have apparently

[2] Bursts of ATM cells will arise as a consequence of the mismatch between some computer data units such as pages, sized as multiples of 1KB, and the ATM payload of 48 bytes.

contiguous access to scattered pages of real memory. This is illustrated in Figure 8.

5 Performance Measurements

In this section, we discuss the performance of the Segmentation and Reassembly hardware. Then we analyze the performance of data transfers across the IBM RISC System/6000 Model 320's implementation of the Micro Channel Architecture.

5.1 Segmentation and Reassembly Hardware

The Segmenter and Reassembler have been fully prototyped with the exception of the electrical to optical network interface. Testing has been accomplished by connecting the Segmenter to the Reassembler in a loop-back configuration via a ribbon cable. With the exception of latency and cell loss due to congestion, this experimental setup reproduces the eventual network environment.

The Segmenter performs as specified in the earlier discussion.

The various stages of the Reassembler also perform as specified in the discussion. Assuming that the Reassembler is not required to service any host requests, the limiting component in the pipeline is the LLM. Since the worst case per cell operation requires 650 ns, and there are 424 bits per cell, the pipeline is capable of processing a burst bandwidth of about 650 Mbps. For sustained operation, this bandwidth would be reduced by up to 50% since the host must also utilize the LLM to drain cells from the reassembly buffer. Even with this reduction in bandwidth, the Reassembler pipeline is still more than capable of support the full bandwidth of an OC-3c connection.

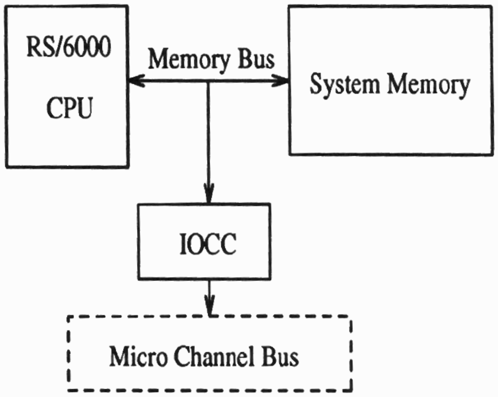


Figure 9: IOCC location in IBM RISC/System 6000

5.2 Micro Channel Architecture Bus Performance

We have carefully studied the performance of data transfers between the host interface and the host's main memory on an RISC System/6000 Model 320.

Using 32 bit streaming transfers, we have found that the bus itself is capable of sustained data transfers at slightly less than 320 Mbps, its peak rate for 32 bit transfers. These data rates were observed card-to-card between peripherals on the Micro Channel bus. Bus arbitration and stream setup time accounted for the deviation from the peak rate.

Unfortunately, when transferring data between the host's main memory and the host interface, significantly lower performance is observed. We determined that the difficulty was with the current implementation of the I/O Channel Controller (IOCC). The IOCC is the connection between the Micro Channel bus and the internal memory bus Figure 9.

To minimize the latency of the host's main memory during a data transfer, the IOCC allocates 16 words of buffering to each transfer channel. Thus, when a word of main memory is read, 16 words of data are loaded into the IOCC's buffers so that consecutive memory accesses are unnecessary.

We have characterized the IOCC's behavior using a logic analysis mainframe with 10 ns resolution connected to the Micro Channel Bus. Between 2 and 3 μs were required to load the IOCC buffer for every 16 words transferred across the bus. The actual data transfer of 16 words requires only 1.8 μs (200 ns for setup and 100 ns per word transferred). This results in a maximum channel efficiency of 44% or 142 Mbps for data transfer between the host interface and the host's main memory.

We expect that later versions of the RISC System/6000 will contain an improved version of the IOCC which will permit a greater utilization of the bandwidth of the Micro Channel bus.

6 Further Architectural Enhance¡ments

The basic architecture presented in this thesis can be easily enhanced to provide increased functionality. This increased functionality can be manifest by the ability of the host interface to provide services such as encryption and/or increased overall performance.

6.1 Support for Encryption

An aspect of wide area networks which has been typically neglected except in the military/national security communities, is cryptographic protection for the data being transferred. While many of the traditional users of the Internet have not been particularily concern about security issues, the next generation of broadband wide area networks will have a much broader base of users, some of whom will be more sensitive about ensuring the privacy and authenticity of their data.

We believe that the host interface provides the best location to integrate per connection encryption into the broadband networking environment. Using a private key encryption scheme such as the Data Encryption Standard (DES) [17][18], which has several high performance hardware implementations [4][31], connection encrytion can be

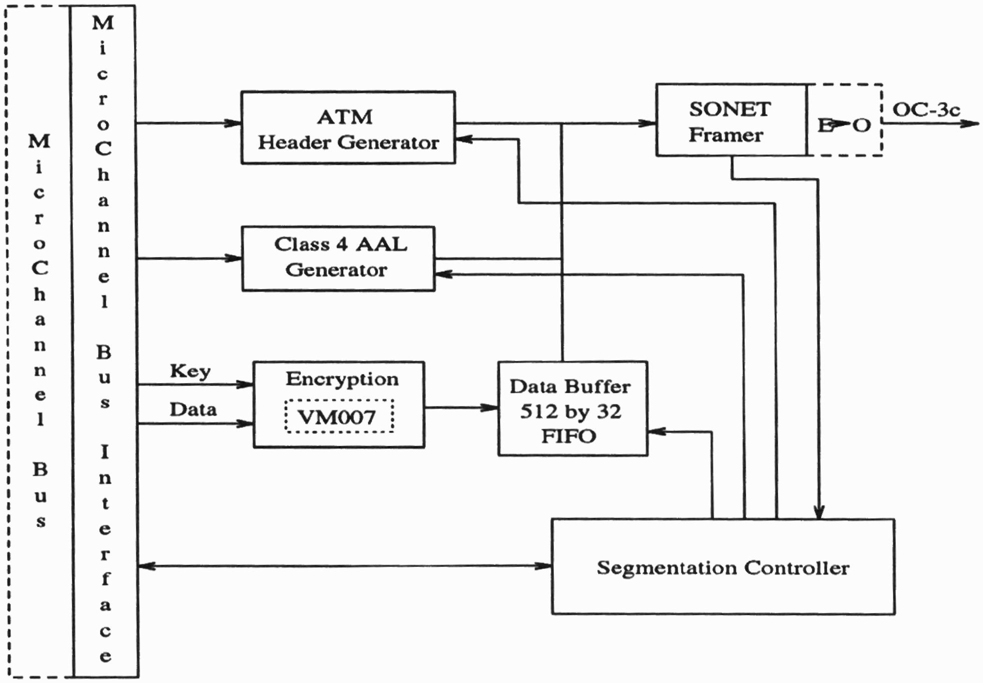


Figure 10: Encryption Support in the Segmenter

made transparently available to hosts. The only role which the host would need to perform is the management and assignment of cryptographic keys.

Assuming a hardware DES implementation such as VLSI Technologies' VM007, which is capable of encrypting/decrypting at 192 Mbps, encryption can be supported by the host interface architecture without adversely affecting overall performance.

For the Segmenter, Figure 10, encryption would be performed on the data as it is moved from the Micro Channel bus to the Segmenter's data buffer. The host would specify the key to be used at the same time that it specifies the VCI and other control information which the Segmenter requires for each block of data. If no encryption is desired, the VM007 can be set in "pass through mode" which does not affect the data.

The changes required to support decryption in the Reassembler are also minimal (Figure 11). The decryption would be performed by the VM007 as the data is moved from the data buffer associated with the Cell Manager into the dual port reassembly buffer

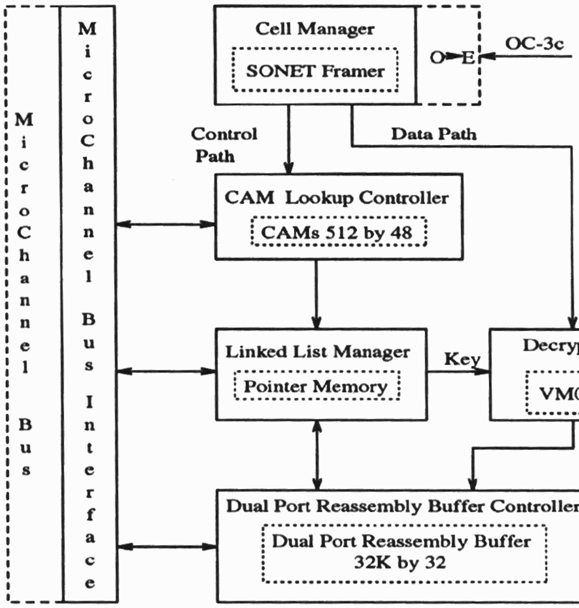


Figure 11: Decryption Support in the Reassembler

(DPRB). The key associated with each connection would be stored in the linked list reassembly datastructure. When data is to be moved into the DPRB, the appropriate key will be loaded into the VM007 at the same time as the pointer to the portion of the DPRB to be used for the data is passed to the Dual Port Reassembly Buffer Controller.

6.2 Support for Higher Bandwidth Network Connections

To take advantage of the high bandwidth of the ATM cell processing pipeline in the Reassembler, multiple Cell Managers can be used to verify the integrity and extract the control fields from incomming cells. Figure 12 illustrates a configuration where four Cell Managers are used to interface to a OC-12 (622 Mbps) network connection. The modifications necessary to support this variation of the basic architecture are minor, as the main change consists of replicating the Cell Manager. To sustain the full OC-12 datarate (622 Mbps), a significantly higher bandwidth path into the RISC System 6000's memory needs to be available. Also, the performance of the LLM would need to be

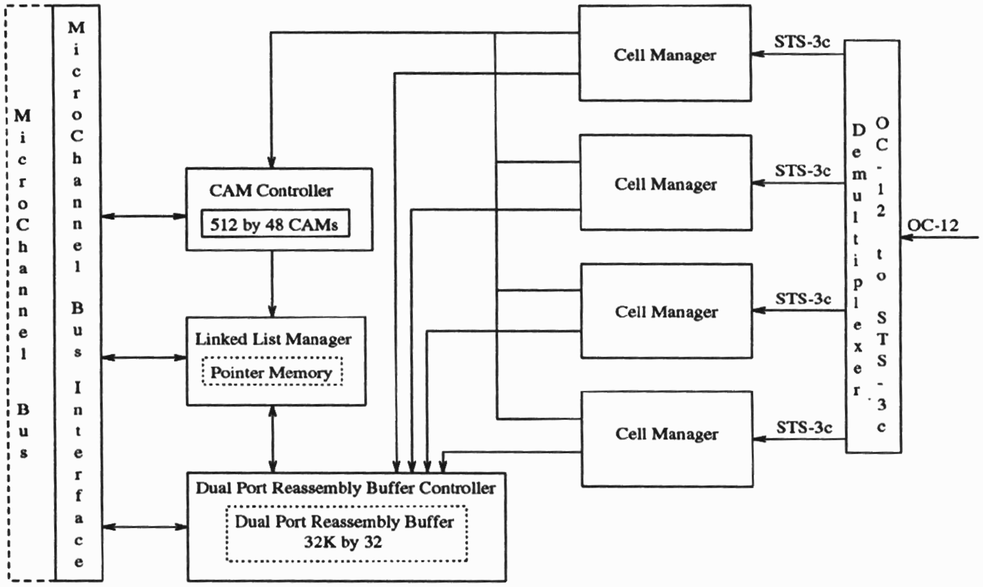


Figure 12: Support for an OC-12 Connection by the Reassembler

improved by either increasing its clock rate or internal parallelism to reduce the number of clock cycles necessary to perform the linked list operations.

Two basic approaches can be taken to increase the performance of the Segmenter. The first is to increase the maximum clock speed of the Segmenter by selecting different implementation technologies. Using either ECL or full custom VLSI the clock rate of could be increased substantially. A second approach which is shown in Figure 13 utilizes components from the existing implementation replicated in parallel. This approach is advantageous not only because it does not require any changes in technology, but also because it provides an easy way to interleave four data streams.

7 Conclusions and a Look to the Future

The hardware and software we have designed and implemented performs remarkably well. The cell manipulation logic on the Host Interface can operate well into the range of 600 Mbps and beyond with minor architectural and/or implementation technologies changes. Our approach of pursuing architectural solutions, such as concurrent operation (as in the

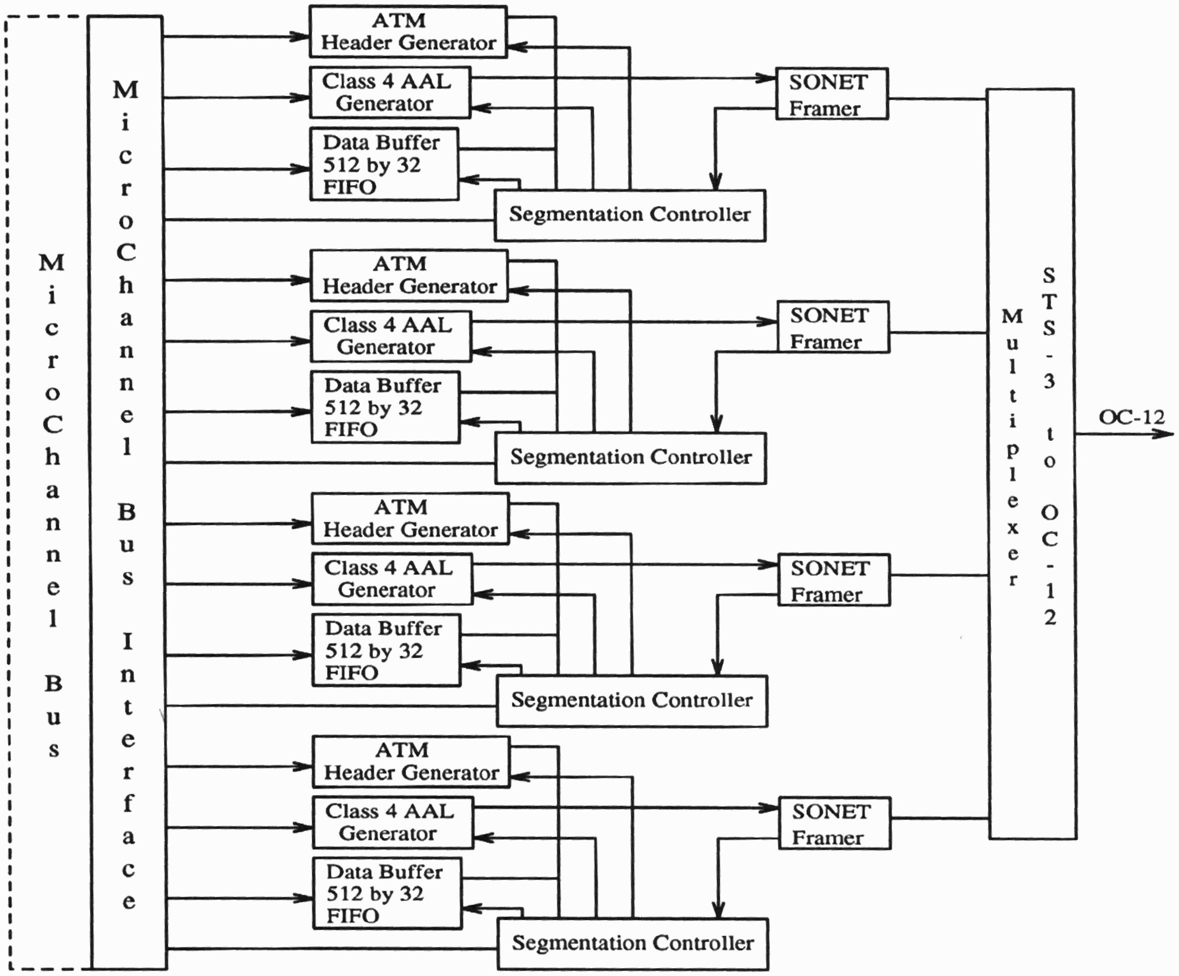


Figure 13: Support for OC-12 Connection by the Segmenter

parallelism in the cell processing pipeline), allows us to take advantage of improvements in technology which would allow higher clock speeds. We were somewhat frustrated in our performance goals by the implementation of the Micro Channel Architecture on the IBM RISC System/6000 Model 320. While the clock rates of the current Micro Channel Architecture could support higher speeds, the current I/O Channel Controller design limits performance to about 140 Mbps. We were surprised to discover this bottleneck, as we expected software or the Micro Channel Architecture bus itself to be the limiting factor. It is hard to blame the designers, as networking at this speed was probably not a consideration in bringing the machine to fruition.

In the near future, we plan to replicate the host interface, and deploy it as a component in the AURORA Testbed. This will allow testing of the interface as a component in a high-speed WAN environment and give us an opportunity to perform protocol processing experiments such as implementations of congestion control strategies. Another possibility which we would like to explore in the context of AURORA is the use of this ATM Interface together with IBM Research's ORBIT [10] card for the RISC System/6000's Micro Channel Architecture. It may be possible to provide a bridge for internetworking PTM and ATM by using these two card together on the bus of an RISC System/6000.

The longer-term research questions raised by this work are centered around workstation architectures. The RISC System/6000, unlike many current-generation workstations, has adequate memory bandwidth to support high-speed networking. The fact that it is not accessible through the Micro Channel bus suggests that perhaps direct-to-memory operations are necessary, with a host interface connected directly to the system memory bus. However, I/O channel architectures such as the Micro Channel Architecture provide a number of attractions, among which are structuring, concurrency control, and features such as virtual address translation with the IOCC. In addition, connection to a bus which is less closely coupled to the CPU can aid portability.

It is unclear how the networking community will resolve its ferocious need for bandwidth, but there seems little question that workstation vendors must provide higher performance access to computational resources and to memory. This performance must be available to attached devices and networks, whether through I/O channels or other attachment schemes.

References

[1] Advance Micro Devices, "Am99C10 256 x 48 Content Addressable Memory," 1989.

[2] Altera Corporation, 1992 Data Book.

[3] H. Abu-Amara, T. Balraj, T. Barzilai, and Y. Yemini, "PSi: A Silicon Complier for Very Fast Protocol Processing," in *Protocols for High Speed Networks*, ed. R. C. Williamson, North-Holland (1989).

[4] Albert G. Broscius and Jonathan M. Smith, "Exploiting Parallelism in Hardware Implementation of the DES," in *Proceedings, CRYPTO 1991 Conference*, Santa Barbara, CA (August, 1991).

[5] Emmanuel A. Arnould, Francois J. Bitz, Eric C. Cooper, Robert D. Sansom, and Peter A. Steenkiste, "The Design of Nectar: A Network Backplane for Heterogeneous Multicomputers," in *Proceedings, ASPLOS-III* (1989), pp. 205-216.

[6] H. B. Bakoglu, G. F. Grohoski, and R. K. Montoye, "The IBM RISC System/6000 Processor: Hardware Overview," *IBM Journal of Research and Development* **34**(1), pp. 12-22 (January, 1990).

[7] Chips and Technologies, "82C611, 82C612 MicroCHIPS: Micro Channel Interface Parts", January 1988.

[8] Israel Cidon and Inder S. Gopal, "PARIS: An Approach to Integrated High-Speed Private Networks," *International Journal of Digital and Analog Cabled Systems* **1**, pp. 77-85 (1988).

[9] D. D. Clark and D. L. Tennenhouse, "Architectural Considerations for a New Generation of Protocols," in *Proc. ACM SIGCOMM '90*, Philadelphia, PA (September 1990).

[10] D.D. Clark, B.S. Davie, D.J. Farber, I.S. Gopal, B.K. Kadaba, W.D. Sincoskie, J.M. Smith, and D.L. Tennenhouse, "An Overview of the AURORA Gigabit Testbed," *Proceedings of the 1992 IEEE Infocom Conference*, Florence, Italy, 1992.

[11] Eric Cooper, Onat Menzilcioglu, Robert Sansom, and Francois Bitz, "Host Interface Design for ATM LANs," in *Proceedings, 16th Conference on Local Computer Networks*, Minneapolis, MN (October 14-17 1991), pp. 247-258.

[12] Eric C. Cooper, Pteenkiste, Robert D. Sansom, and Brian D. Zill, " Protocol Implementation on the Nectar Communications Processor," in *Proceedings, SIGCOMM '90*, Philadelphia, PA (September 24-27, 1990), pp. 135-144.

[13] IBM Corporation, *IBM RISC System/6000 POWERstation and POWERserver: Hardware Technical Reference, Micro Channel Architecture*, IBM Order Number SA23-2647-00, 1990.

[14] IBM Corporation, *IBM RISC System/6000 POWERstation and POWERserver: Hardware Technical Reference, General Information Manual*, IBM Order Number SA23-2643-00, 1990.

[15] Bruce S. Davie, "Host Interface Design for Experimental, Very High Speed Networks," in *Proceedings, Compcon Spring '90*, San Francisco, CA (February 1990), pp. 102-106.

[16] Bruce S. Davie, "A Host-Network Interface Architecture for ATM," in *Proceedings, SIGCOMM 1991*, Zurich, Switzerland (September 4-6, 1991), pp. 307-315.

[17] Federal Information Processing Standard #46: The Data Encryption Standard, National Bureau of Standards Information and Computer Systems Technology Division.

[18] Federal Information Processing Standard #81: Operational Modes of the Data Encryption Standard, National Bureau of Standards Information and Computer Systems Technology Division.

[19] J. Giacopelli, J. Hickey, W. Marcus, W. D. Sincoskie, and M. Littlewood, "Sunshine: A High-Performance Self-Routing Broadband Packet Switch Architecture," *IEEE Journal on Selected Areas in Communications* **9**(8), pp. 1289-1298 (October, 1991).

[20] David J. Greaves, Dimitris Lioupis, and Andy Hopper, "The Cambridge Backbone Ring," in *Proceedings, INFOCOM 1990 Conference* (1990). (also Olivetti Research Laboratory Technical Report 90/2)

[21] CCITT Recommendation I.361, *ATM Layer Specification for B-ISDN*, 1990.

[22] CCITT Recommendation I.363, *B-ISDN ATM Adaptation Layer (AAL) Specification*, 1990.

[23] H. Kanakia and D. Cheriton, "The VMP Network Adapter Board (NAB): High Perfromance Network Communication for Multiprocessors," in *Proceedings, SIGMETRICS '88* (1988).

[24] Jeffrey C. Mogul and Anita Borg, "The Effect of Context Switches on Cache Performance," in *Proceedings, Fourth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-IV)*, Santa Clara, CA (April 8-11, 1991), pp. 75-85.

[25] Thomas J. Robe and Kenneth A. Walsh, "A SONET STS-3c User-Network Interface IC," in *Proceedings, Custom Integrated Circuits Conference*, San Diego, CA (May, 1991).

[26] Computer Staff, "Gigabit Network Testbeds," *IEEE Computer* **23**(9), pp. 77-80 (September, 1990).

[27] Andrew S. Tannenbaum, "Computer Networks," Prentice Hall, Second Edition, 1988.

[28] K. L. Thompson, "UNIX Implementation," *The Bell System Technical Journal* **57**(6, Part 2), pp. 1931-1946 (July- August 1978).

[29] C. B. S. Traw and J. M. Smith, "A High-Performance Host Interface for ATM Networks," *Proceedings ACM SIGCOMM '91*, Zurich, September 1991.

[30] C. B. S. Traw and J. M. Smith, "Implementation and Performance of an ATM Host Interface for Workstations," in *Proceedings, IEEE Workshop on the Architecture and Implementation of High-Performance Communications Subsystems (HPCS '92)*, Tucson, AZ (February 17-19, 1992).

[31] VLSI Technologies, "Advanced Information: VM007 Data Encryption Processor," 1991.

[32] Martina Zitterbart, "High-Speed Transport Components," *IEEE Network*, pp. 54-63 (January, 1991).

A Appendix: Glossary of Terms

AAL	ATM Adaptation Layer
AAL4	ATM Adaptation Layer Class 4
AIX	Advanced Interactive Executive
ATM	Asynchronous Transfer Mode
CAM	Content Addressable Memory
CCITT	Comite Consultatif Internationale de Telegraphique et Telephonique
CLC	CAM Lookup Controller
CRC	Cyclic Redandence Check
CS-PDU	Convergence Sublayer Protocol Data Unit
DARPA	Defense Advanced Research Projects Agency
DMA	Direct Memory Access
DPRB	Dual Port Reassembly Buffer
DPRBC	Dual Port Reassembly Buffer Controller
ECL	Emitter Coupled Logic
FIFO	First In First Out
Gbps	Gigabit per second
IOCC	I/O Channel Controller
ISO	International Standardization Organization
LLM	Linked List Manager
Mbps	Megabits per second
MID	Multiplex Identifier
ns	Nanosecond
NSF	National Science Foundation
OC-3	Optical Carrier (155 Mbps)
OC-3c	Optical Carrier (155 Mbps – concatenated payload)
OC-12	Optical Carrier (622 Mbps)
OSI	Open Systems Interconnection
PFT	Page Frame Table
RAM	Random Access Memory
SONET	Synchronous Optical Network
STS-3	Synchronous Transport System (155 Mbps)
STS-3c	Synchronous Transport System (155 Mbps – concatenated payload)
TCW	Translation Control Word
μ**s**	Microsecond
VCI	Virtual Circuit Identifier

VLSI	Very Large Scale Integration
VPI	Virtual Path Identifier