Report Number: WUCS-92-36

1992-11-01

Simulation of an ATM-FDDI Gateway

Milind M. Buddhikot, Sanjay Kapoor, and Gurudatta M. Parulkar

Asynchronous transfer mode (ATM) networks can support a wide variety of applications with varying data rates. Fiber Distributed Data Interface (FDDI) networks are targeted to support similar applications, but only in a LAN environment. Both of these networks are characterized by high data rates and low error rates. In addition, they can provide applications with varying degrees of performance guarantees. A problem of designing a high-performance gateway architecture for internetworking between these two important classes of networks was addressed in [Kapoor, 1991b]. An important part of the gateway design philosophy was to partition the internet functions into critical and... **Read complete abstract on page 2.**

Simulation of an ATM-FDDI Gateway

Milind M. Buddhikot, Sanjay Kapoor, and Gurudatta M. Parulkar

Complete Abstract:

Asynchronous transfer mode (ATM) networks can support a wide variety of applications with varying data rates. Fiber Distributed Data Interface (FDDI) networks are targeted to support similar applications, but only in a LAN environment. Both of these networks are characterized by high data rates and low error rates. In addition, they can provide applications with varying degrees of performance guarantees. A problem of designing a high-performance gateway architecture for internetworking between these two important classes of networks was addressed in [Kapoor, 1991b]. An important part of the gateway design philosophy was to partition the internet functions into critical and non-critical paths. The critical path consists of per packet processing, an is implemented in hardware. The non-critical path consists of connection, resource, and route management, and is implemented in software. In order to establish the feasibility of this approach to the design of future high-speed gateways, an extensive simulation of the prosed gateway architecture was carried out. The primary focus of this paper is on the description of the simulation model, and its use in characterizing the performance of the proposed gateway architecture.

Simulation of an ATM-FDDI Gateway

Milind M. Buddhikot, Sanjay Kapoor and Gurudatta M. Parulkar

WUCS-92-36

November 1992

Department of Computer Science
Washington University
Campus Box 1045
One Brookings Drive
St. Louis MO 63130-4899

Simulation of an ATM–FDDI Gateway*

Milind M. Buddhikot
Department of Computer Scince,
Washington University in St. Louis,
St. Louis, MO 63130.
milind@dworkin.wustl.edu

Sanjay Kapoor
Department of Electrical Engineering,
Washington University in St. Louis,
St. Louis, MO 63130.
kapoor@wueel.wustl.edu

Gurudatta M. Parulkar
Department of Computer Scince,
Washington University in St. Louis,
St. Louis, MO 63130.
guru@flora.wustl.edu

ABSTRACT

Asynchronous transfer mode (ATM) networks can support a wide variety of applications with varying data rates. Fiber Distributed Data Interface (FDDI) networks are targeted to support similar applications, but only in a LAN environment. Both of these networks are characterized by high data rates and low error rates. In addition, they can provide applications with varying degrees of performance guarantees. A problem of designing a high-performance gateway architecture for internetworking between these two important classes of networks was addressed in[Kapoor, 1991b]. An important part of the gateway design philosophy was to partition the internet functions into critical and non-critical paths. The critical path consists of per packet processing, and is implemented in hardware. The non-critical path consists of connection, resource, and route management, and is implemented in software. In order to establish the feasibility of this approach to the design of future high-speed gateways, an extensive simulation of the proposed gateway architecture was carried out. The primary focus of this paper is on the description of the simulation model, and its use in characterizing the performance of the proposed gateway architecture.

*This work was supported in part by the National Science Foundation, and an industrial consortium of Ascom Timeplex, Bellcore, BNR, DEC, Italtel SIT, NEC, NTT, and SynOptics.

1 Introduction

The existing computer communication environment can be characterized as an internetwork consisting of many low speed networks interconnected by gateways. We believe that the future communications environment will continue to be an internetwork. However, it will include emerging high-speed networks such as ATM and FDDI , and will need to support a wide variety of applications. These applications will typically require high bandwidth and performance guarantees; examples include video distribution, multimedia conferencing, and interactive remote visualization. In order to meet these demands, new protocols and architectures will have to be developed. As a first step in this direction, we have proposed a novel internetwork abstraction called the Very High Speed Internetwork Abstraction (VHSI) [Parulkar, 1990].

The VHSI, like the existing Internet, will include a variety of local, regional, and national networks. These networks will include public networks, private corporate networks, and networks supported by federal agencies. An important component of the VHSI abstraction is the Multipoint Congram-oriented High performance Internet Protocol (MCHIP). Higher level protocols use MCHIP to communicate over the internetwork and to obtain the desired levels of performance guarantees. Another important aspect of the VHSI abstraction, which is of special relevance to this paper, is its high speed and high-performance gateway architecture. Other components of VHSI include resource management servers, internetwork routing servers, and interfaces to network access protocols.

The purpose of a gateway in VHSI is to interconnect disparate networks and to implement resource reservation and internetwork routing policies. In designing the architecture of VHSI gateways, we have decided to emphasize the policies rather than the mechanisms employed. In order to fully illustrate these policies, we chose to undertake the actual design and evaluation of a gateway architecture that would interconnect two widely diverse high-performance networks. For this purpose, we chose to design a gateway interconnecting ATM and FDDI networks. This choice was dictated not only by the fact that these networks have vastly diverse characteristics, but also by our belief that they are typical of the classes of networks that will find widespread use in the computer communication networks of the future.

An important part of our gateway design philosophy is to partition the data transport functions into critical and non-critical paths. The critical path consists of per-packet processing, and would typically be implemented in hardware for reasons of speed and efficiency. The non-critical path consists of connection, resource, and route management. This path would typically be implemented in software for reasons of flexibility, and since most of the functions comprising this path require the ability to do fine tuning. In contrast to this approach, most of the existing gateways implement all their functions in software, thus sacrificing both speed and efficiency.

This paper describes a simulation of a two-port ATM-FDDI gateway. The detailed design of the gateway has been described in [Kapoor, 1991a]. The primary objective in undertaking this extensive simulation study was to

characterize the gateway performance under various input traffic conditions and design parameters. This would help in supporting our design policies, primarily the philosophy of implementing the critical path in hardware and the non-critical path in software. Another objective of the simulation was to perform an extensive functional verification of our design.

The outline of the rest of the paper is as follows: Section 2 gives a brief description of the gateway architecture design that is the target of our simulation model. Section 3 focuses on the specific performance metrics that we were interested in, and the reasons for selecting these metrics. Section 4, presents a brief overview of the simulation tool used in developing the simulation model. Section 5 focuses on the simulation model itself, while Section 6 describes the various experiments conducted and the results obtained from these experiments. Finally, section 7 provides conclusions.

2 Architecture Overview

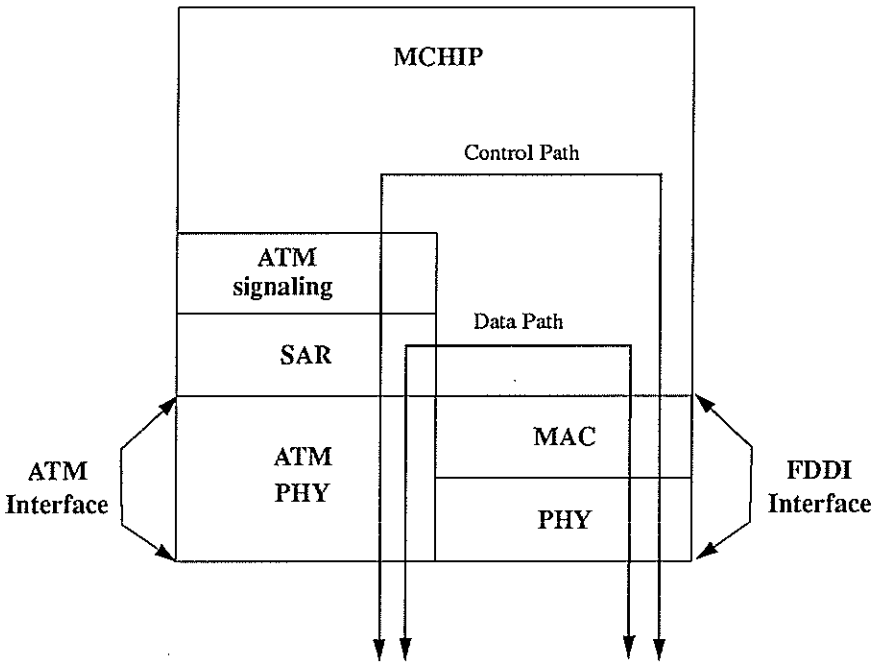

Figure 1: Protocol Structure in a Gateway

This section provides a brief description of the ATM-FDDI gateway design. Further architectural and design details, the are contained in the original design paper [Kapoor, 1991a]. Figure 1 is a high-level view of the layering and protocol structure upon which the gateway design is based. In addition to the ATM and FDDI physical interfaces, the figure also shows the ATM segmentation and reassembly layer (which implements the SAR protocol [Escobar, 1991]), the ATM signalling layer, the FDDI media access (MAC) sublayer, and the internet layer protocol (in this case, MCHIP). Also shown in the figure are trajectories that show the layers traversed by data and control information as it passes through the gateway; we shall refer to these as the data and control paths

respectively.

A simplified block diagram of the gateway hardware architecture appears in Figure 2. Note that the non-critical (control) path functions are implemented in software run on a processor called the Node Processing Element (NPE). All the critical (per cell or per frame) functions are implemented in hardware using two custom VLSI chips and two commercially available chips. The rest of this section summarizes the purpose and operation of each of the blocks shown in the Figure 2.

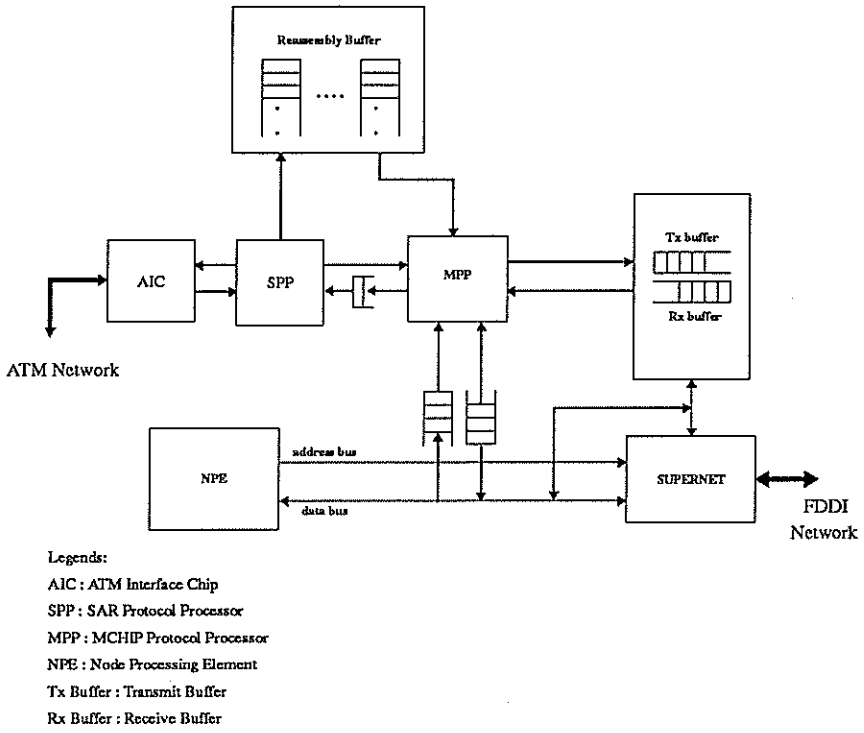

Figure 2: A Two Port ATM–FDDI Gateway

ATM Interface Chip (AIC): The AIC implements the ATM physical layer protocol and provides an interface to the ATM network. The AIC is described in detail in Turner's ATM architecture[Turner, 1988], where it is referred to as processor PP1.

SAR Protocol Processor (SPP): The SPP implements the SAR protocol, which is responsible for segmenting data and control frames from higher level protocols into ATM cells, and for reassembling these frames upon the receipt of all the constituent cells. Part of the job relegated to the SPP involves reassembly buffer and timer management; for this purpose, it interfaces to the AIC, MPP, and reassembly buffer memory. Figure 3 shows the translation phases that a typical ATM cell undergoes when processed by the SPP.

MCHIP Protocol Processor (MPP): The job of the MPP is to route reassembled FDDI frames or ATM cell

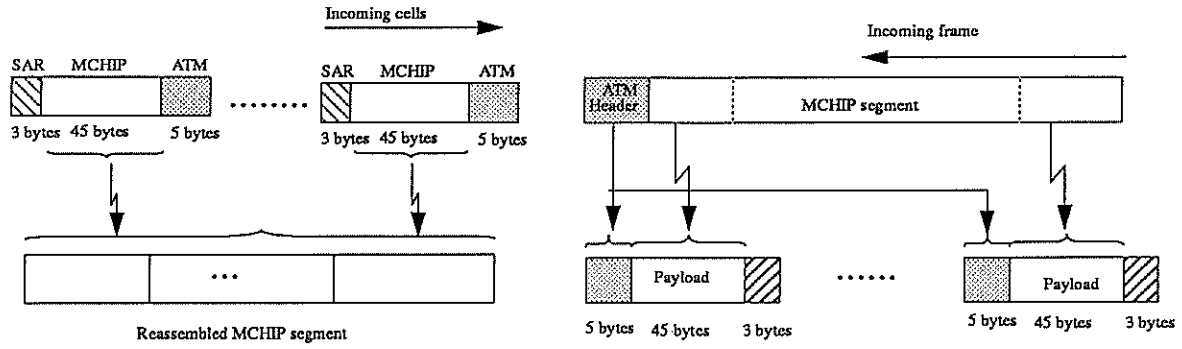

Figure 3: Packet Translations in SPP

fragments to the appropriate destination based upon the type of the cell/frame and information contained in the MCHIP routing tables. The MPP is also responsible for detecting incoming control frames and handing them over to the NPE, as well as accepting and forwarding control frames from the NPE.

Upon receipt of a reassembled frame from the SPP, the MPP decodes the frame type field from the header. If this field indicates that the frame is an ATM or MCHIP control frame, it forwards the frame directly to the NPE. If, on the other hand, the frame contains data, a table lookup is performed using MCHIP ICN to determine the destination FDDI address. Following this, FDDI-specific headers are prepended to the frame before writing it into the transmit buffer memory.

In the reverse direction, the MPP reads the next frame from the receive buffer memory. It then performs a table lookup using the Internet Channel Number (ICN) field of the MCHIP header. This table lookup yields an ATM header that is appended to the frame, which is then forwarded to the SPP through a FIFO.

Node Processing Element (NPE): The NPE is a general purpose microprocessor running software implementations of the ATM signaling protocol, FDDI connection and station management, and MCHIP congram management. The NPE also performs various housekeeping functions for the gateway hardware, such as processing interrupts, initializing hardware components, and configuring the synchronous and asynchronous queues in the transmit and receive buffer memory of the SUPERNET chip set.

SUPERNET [1]: The SUPERNET chip set implements the physical and media access FDDI protocols in VLSI [AMD, 1990]. Station and connection management are not implemented in the SUPERNET chip set. However, it does provide several registers to keep track of ring statistics and can be used by the station management software running on the (NPE).

Buffer Memories: There are three distinct buffer memories in the gateway: the reassembly buffer memory, the transmit buffer memory, and the receive buffer memory. The maximum size of transmit and receive buffers

is dictated by the SUPERNET specifications. In our design of the gateway, we allow for reassembly of a maximum of two full size FDDI frames per active connection. Hence, using a 256K reassembly buffer would allow support for 32 connections.

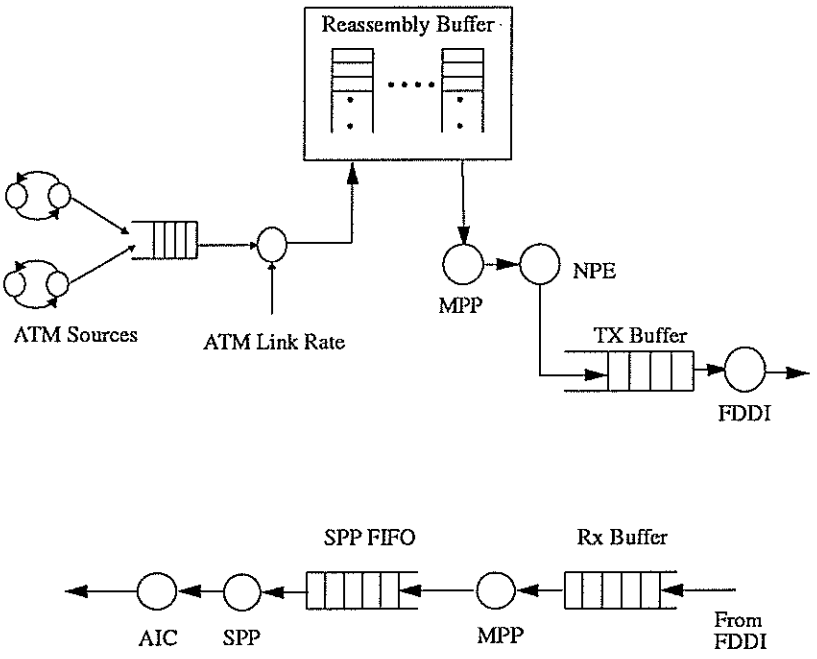

Figure 4: Queueing in the Gateway Architecture

Figure 4 shows two simple queueing systems that represent data flow in either direction (ATM to FDDI and vice-versa) in the gateway. For the ATM to FDDI direction,here after referred to as forward direction, the main points at which queueing occurs are the reassembly buffer and the transmit buffer. The service mechanisms for these buffers are complex. The reassembly buffer is serviced by a tandem server formed by the MPP and NPE. This service mechanism depends on the state of the transmit buffer, such as the availability of the buffer space and thus is not work conserving. On the other hand, the transmit buffer in the FDDI interface is serviced by the timed-token-rotation mechanism of FDDI. It can be seen that the queueing and service mechanisms are quite complex and are potential sources of performance bottlenecks. In the FDDI to ATM direction, here after referred to as the reverse direction, the receive buffer is serviced by the MPP. This is a much simpler service discipline and therefore, there are no potential bottlenecks in the reverse direction.

3 Performance Metrics

In this section, we shall present a set of performance metrics that can be used to characterize the gateway performance. These metrics will serve as "outputs" for our simulation study. The choice of these metrics also served to guide us in the design and implementation phases of the actual simulation. Enumerated below are the proposed metrics, along with brief descriptions of each:

Load vs Throughput performance of the Gateway: The load-throughput behavior of the gateway characterizes what fraction of the offered load can be successfully carried by the gateway.

Cell loss probability in SPP: The SPP drops cells if the reassembly buffer is full, or if the arrival rate is higher than the service rate (that is, the NPE is too slow).

Frame loss probability in MPP: Frames that have not been completely reassembled by the SPP are of no use to higher level protocols or the end user, and are therefore dropped in the MPP.

Latency of completely assembled frames (through the gateway): This is the time spent in the gateway by the first constituent cell of a frame that is successfully reassembled within the gateway; it is a measure of the delay performance of the gateway architecture.

The performance metrics described above strongly depend on design parameters such as the sizes of the transmit, receive, and reassembly buffers, the processing speed of the NPE, and the maximum number of active connections supported by the gateway. In addition, they are influenced by changes in input traffic patterns. A careful analysis of the effects of these parameters on the metrics can cast light on design tradeoffs, and enable us to (over) engineer the gateway hardware so as to obtain acceptable levels of performance.

The queueing and service mechanisms in the gateway architecture are very complex and an analytical performance model to characterize performance metrics seems mathematically intractable. On the other hand, a detailed function level simulation study can be done with ease. Such a study can also be used to do a extensive functional verification of the design. A graphical general purpose simulation tool called BONES[2], allows typical architectural components like multiplexors, memories, buffers, timers and processors to be modeled easily and hence is useful in developing functional model for gateway architecture.

4 Block Oriented Network Simulator [BONeS]

The gateway simulation model was developed using a commercially available interactive X window based graphical simulation tool called BONeS. This tool requires the user to construct the desired model using two primitives:

[2]BONeS (Block Oriented Network Simulator) is a trademark of Comdisco Systems, Inc.

blocks and data structures. Blocks can be constructed from other existing blocks by providing appropriate interconnections; this is much like creating a block diagram description of a system. BONES provides a set of rudimentary blocks that are used to ultimately create all other blocks; thus, a hierarchically organized description of blocks can be developed that completely specifies the system being simulated. Blocks process data such as traffic formats, packets, and messages; these entities are represented by data structures. BONES provides a graphical interface that can be used to construct block diagrams interactively and to specify the relevant data structures. Facilities are also provided to insert probes that can gather statistics at various points in the system. The final code produced by BONES is in C; this is compiled by the local C compiler before execution.

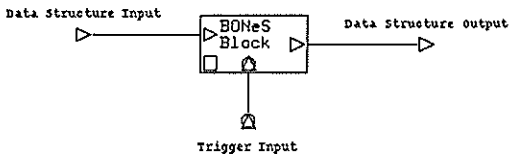

Figure 5: BONES block

A typical BONES block is shown in Figure 5; blocks usually have inputs, outputs and a set of parameters associated with them. The topmost level block, called a system, does not have any inputs or outputs. Parameters associated with BONES blocks can be exported to higher levels for use by blocks in that level, or localized so that their use is restricted to the scope of the defining block.

atm_ds Data Structure	
Field	Type
VCI	Integer
Length	Integer
Time Created	Real
Data	Root-object
Last Time Stamped	Real
SAR sequence number	Integer
SAR F bit	Integer
SAR C bit	Integer

Table 1: Cell Data Structure

As an example of data structure usage in BONES, Table 1 shows the ATM cell data structure used in our simulation. Note that in addition to fields corresponding to header and trailer information in a cell, these data structures also contain other fields such as `Time Created`, `Last Time Stamped` that are used in the simulation to capture various statistics.

5 Simulation Overview

This section, presents our simulation model of the gateway architecture. The model is described using the BONeS blocks, we developed in the course of this simulation.

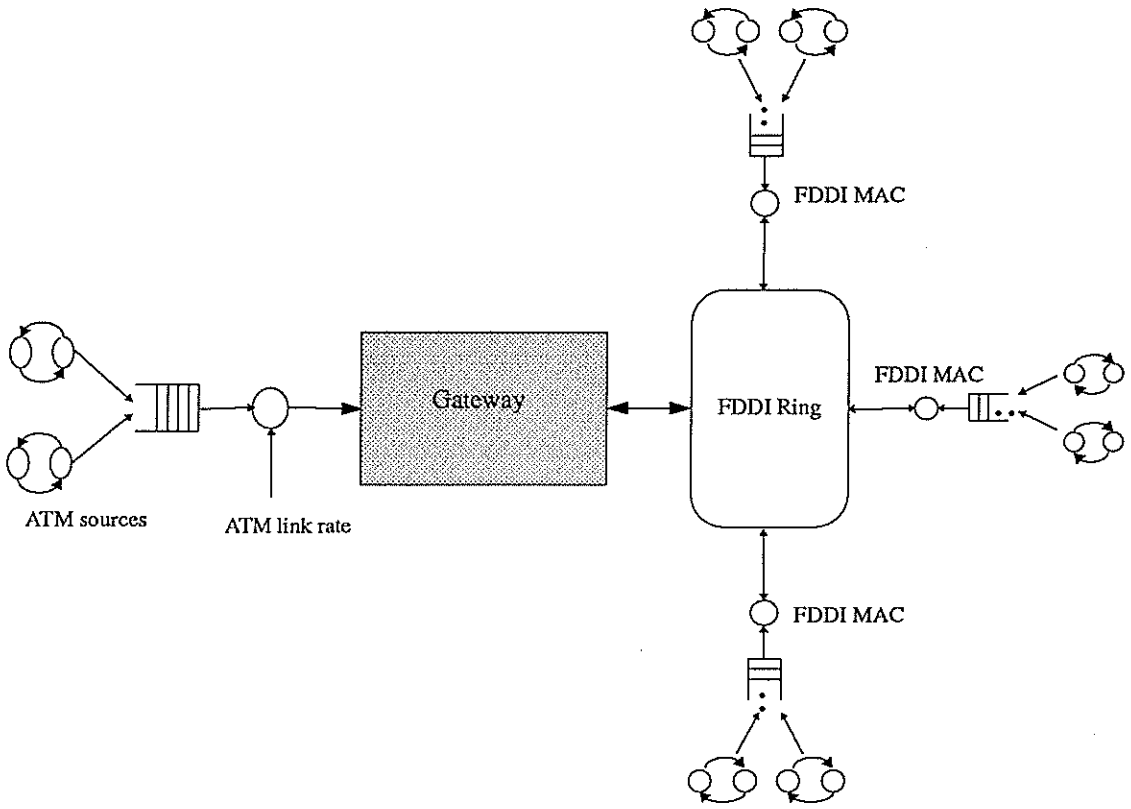

Figure 6: Simulation Definition

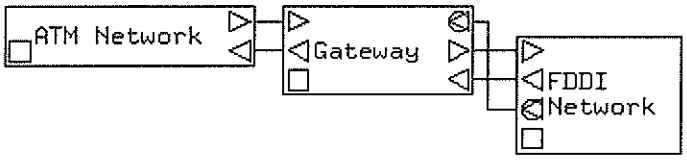

Figure 7: System Model

Figure 6 shows a high level view of the system. The corresponding BONeS system block diagram appears in Figure 7. Note that the gateway block is pictured between a model of an ATM network and a module that simulates FDDI network traffic. The latter module is part of the BONeS standard distribution. The ATM network is simulated as a set of bursty sources (representing independent connections) feeding an infinite length queue which is serviced at the ATM link rate of 155 Mbps. The particular service discipline used is First In First Out (FIFO). The gateway model itself is constructed as a hierarchy of seven layers of BONeS blocks. Descent through this hierarchy corresponds to an increase in the perceived degree of detail. An effort was made to maintain a

close resemblance between BONeS blocks and the hardware components comprising the gateway; this will become apparent in the following paragraphs, where we attempt to describe individual blocks in the system.

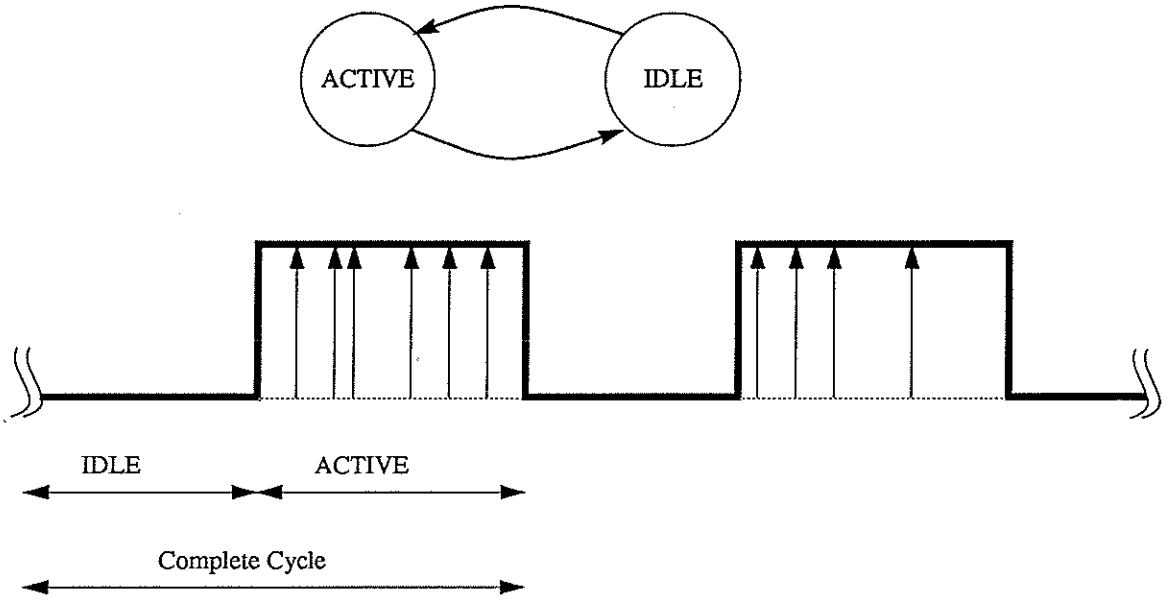

Figure 8: Traffic Model

As mentioned earlier, the ATM network is modeled as a set of bursty sources feeding an infinite length queue, which is FIFO serviced at 155 Mbps. Each of these sources represents the network traffic on an active ATM connection. A two state Markov chain is used to statistically describe each source, as shown in Figure 8. The sources alternate between an active state and an idle state, with the time in each state being exponentially distributed. During the active state, sources generate Poisson traffic, and during the idle state, no traffic is generated. A three tuple $< \Lambda_p, \Lambda_a, B >$ can be used to fully characterize the bursty nature of a source. Here, Λ_p is the peak bandwidth, Λ_a is the average bandwidth, and B is a burst factor that provides a measure of the amount by which the average bandwidth is exceeded during an active period.

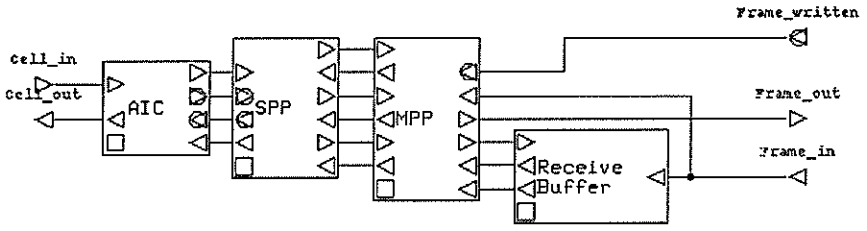

Figure 9: Gateway Model

Figure 9 shows the next lower level of BONeS blocks comprising the gateway block in Figure 7. The SPP and the MPP have been explicitly modeled as BONeS blocks, as have the ATM interface chip and the receive buffer memories. Note that no control path traffic has been modeled; it is assumed that control traffic does not interfere

with data path processing. The NPE has been modeled as a processing delay, placed in SPP-MPP acknowledgment path. Finally, note that DMA access to the transmit buffer has not been explicitly modeled in our system.

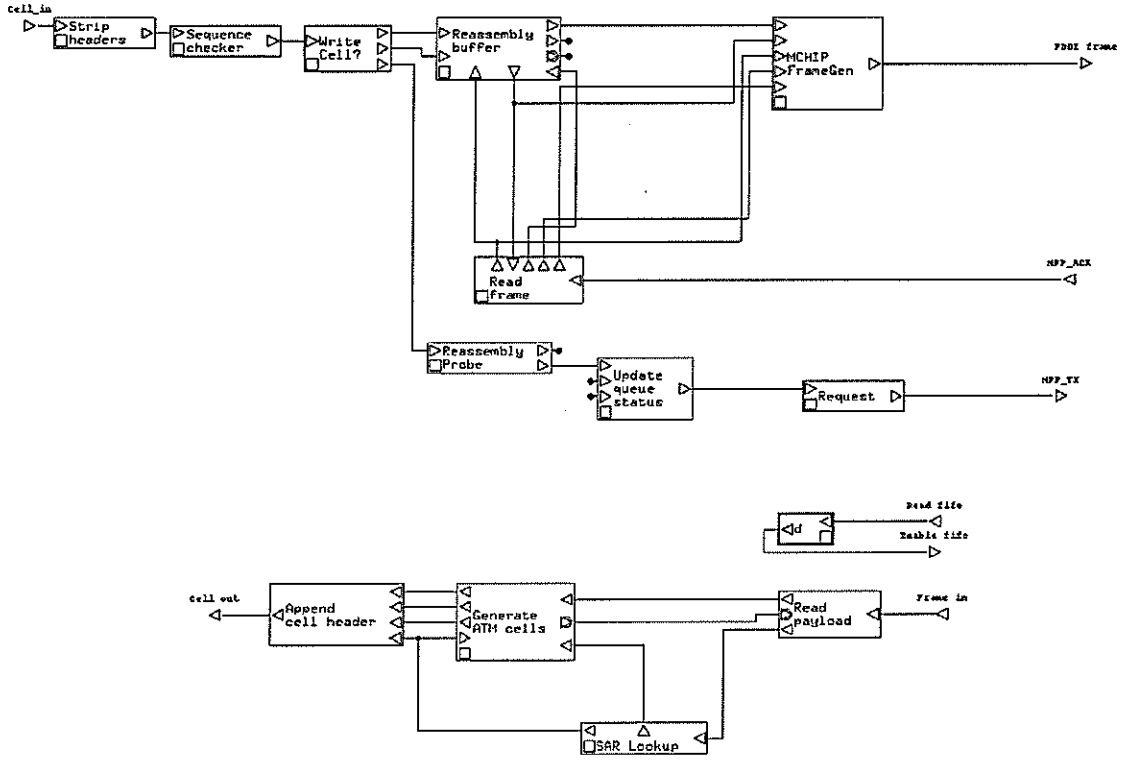

Figure 10: SPP Model

Figure 10 shows the BONeS block diagrams for two pipelines that comprise the system's view of the SPP. One of these pipelines models data flowing from the ATM interface to the FDDI transmit buffers, while the other pipeline models data flow in the reverse direction (that is, from the FDDI receive buffers to the AIC). Note that the function of these pipelines is the same as that of the two hardware pipelines that are required in the VLSI implementation of the SPP. The `Strip Headers`, `Sequence Checker` and `Write Cell?` blocks collectively process the incoming cell data structure and decide if it should be dropped or reassembled in an appropriate queue. The `Reassembly Buffer` block uses the queue number information provided by the above blocks to write to an appropriate queue. The `Update queue status` block processes each cell data structure, to update appropriate state tables (which are described in detail below) and generates read request for the MPP model via `Request` block. The `Read Frame` clock uses acknowledgement from MPP model, to simulate reading reassembled frame and generating appropriate MCHIP segment data structure through `MCHIP FrameGen` block.

In the reverse direction, the `Generate ATM cells` block uses the SAR information provided by the `SAR Lookup` block and the payload information provided by the `Read Payload` block to simulate fragmentation of FDDI frames into ATM cells.

It is instructive to list the state tables that are used for lookups during the operation of these pipelines. In particular, we list below the state tables that are maintained on a per-connection basis for operation of the ATM-FDDI pipeline. Note that this pipeline is responsible for the reassembly of FDDI frames from incoming ATM cells.

- The **Queue Number** state table stores the identity of the reassembly queue currently being used for each active connection. Thus, the VCI field from an incoming ATM cell can be used to locate the appropriate reassembly queue for the connection to which that cell belongs.

- The **Queue Status** state table stores the status of the queues allocated to every active connection. Entries in this table indicate availability of reassembly queues. The table entry corresponding to a given queue number can have one of two values: idle or busy. A queue is busy if it contains a completely reassembled frame, while it is considered idle if it is empty or if it contains a partially reassembled frame.

- The **Frame Status** state table maintains the status of the reassembled frame in each reassembly queue. Entries in this table are accessed using the queue number. A frame may have missing cells and therefore be in error; hence, frame status table entries can have one of two values: NoError and Inerror. The frame status information is forwarded to and used by the MPP to determine if the frame should be dropped or forwarded.

- The **Expected Sequence Number** state table stores the SAR sequence number of the next cell that the SPP expects to receive on each active connection. These table entries are accessed using the VCI field from incoming ATM cell headers.

- The **Start Sequence Number** state table stores the sequence number of the first cell in the segment next to the one currently being assembled for every active connection. The entries in this table are accessed using the VCI field from incoming ATM cells. The information in this table is used by the SPP to handle cases where the cell with framing [F] bit is dropped or lost.

Note that these state tables are not created or maintained by any one particular block in Figure 10, but are used and modified collectively by all blocks. Also, note that most of these tables are implemented as integer vector memories.

Just as the SPP was modeled as two pipelines, one for each direction of data flow, the MPP is also modeled by two pipelines. The BONeS block diagrams for these pipelines are shown in Figure 11. The top part of the figure is the forward pipeline, which is responsible for reading reassembled frames from the reassembly buffer and writing them to the transmit buffer. The `Node Processor` block models both the NPE and the delay incurred in copying the frame. The MPP `Handshake` block is responsible for the handshaking between the SPP and the MPP. The `Drop`

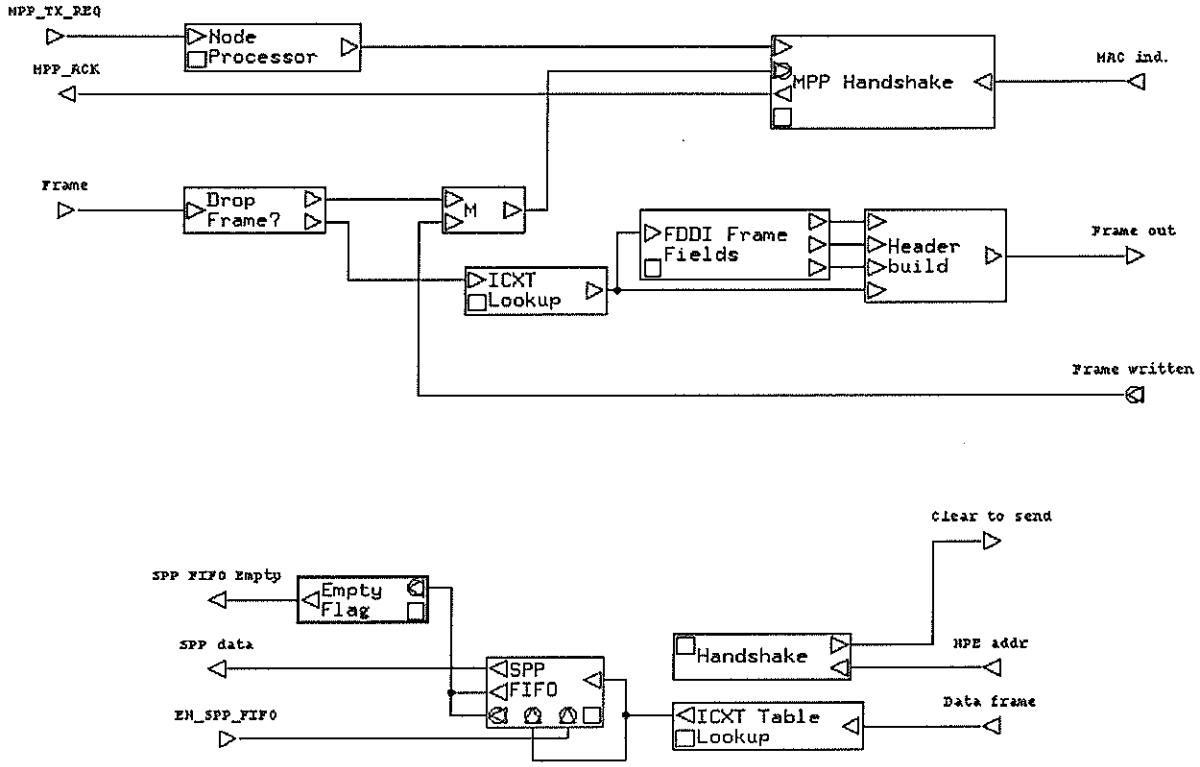

Figure 11: MPP Model

`Frame?` module discards MCHIP segments marked Inerror, and forwards the remaining segments to the channel translation block (`ICXT Lookup`). Subsequently, the `FDDI Frame Fields` and `Header Build` blocks insert header fields to construct a FDDI frame that is ready to be sent. The lower part of Figure 11 shows the pipeline in the reverse direction. Here, the `ICXT Table Lookup` block uses the source address and ICN fields from incoming FDDI frames to perform a table lookup that yields the appropriate ATM header. Note that the `SPP FIFO` block simulates the FIFO between the SPP and the MPP.

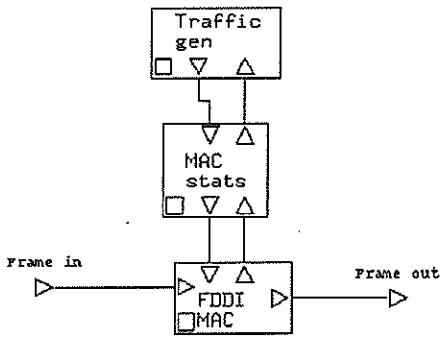

Figure 12: FDDI MAC

The FDDI network block used in the simulation belongs to a library of local area network modules provided by BONeS [Comdisco, 1991b]. This block models a set of nodes on an FDDI ring, with one of the nodes serving as a gateway. Figure 12 shows the BONeS block diagram of a typical node in this layout. The traffic generator block shown in this figure is capable of generating synchronous traffic as well as asynchronous prioritized traffic. The FDDI MAC block implements the FDDI media access protocol. For the purposes of our simulation, we modified this block on the gateway node to reflect the handshaking between the SPP and the MPP. We assume that the FDDI network carries only connection oriented traffic. With this assumption, the traffic generator in every node needs to maintain a translation table of ICN vs. destination node address for each active connection. We modified the traffic generator in each node to have this feature and generate bursty traffic. Furthermore, we have made the assumption that all the traffic is asynchronous [Katz, 1992]. The implication of this assumption is that performance results obtained are optimistic.

6 Simulation Experiments

A number of experiments were conducted using the simulation; these experiments fall into two basic categories: functional verification experiments and performance evaluation experiments. In this paper, we shall not present any results from the former category due to lack of space. We shall, however, venture to say that these experiments were immensely useful in verifying the correctness of the design as well as correctness of different parts of the simulation, such as the operation of various table translations, the reassembly and fragmentation algorithms and the SPP - MPP handshake mechanism. The performance evaluation experiments can be classified as follows:

- Effect of input ATM traffic patterns.
- Effect of buffer size constraints.
- Effect of the NPE delay.
- Effect of FDDI parameters.
- Effect of the number of active connections.

For these experiments, some parameters were always maintained constant. Table 2 lists nine parameters that remained constant throughout these experiments and their values. All experiments study the behavior of the gateway hardware, which is largely independent of the number of stations on the FDDI network and therefore, all the experiments use a fixed four node FDDI network.

Parameter	Value
Number of Active Connections	8
Clock Cycle	25 ns
Maximum FDDI frame size	4096 bytes
ATM Link capacity	150 Mbps
FDDI Link Capacity	100 Mbps
Node1 priority	7
Node2 priority	7
Node3 priority	7
Gateway priority	7
TTRT	50 ms

Table 2: FDDI Network Parameters

6.1 Effect of Burst Factor

Recall that the burst factor is a measure of how much extra traffic the source generates over and above its average bandwidth when it goes into the active state. A higher burst factor makes the active and idle periods of a source longer. Longer active periods demand larger buffers on a transient basis. Therefore, at very high burst factors, the gateway is expected to experience a high cell/frame loss.

Parameter	Value
Transmit Buffer Size	16 Maximum size FDDI frames
Reassembly Buffer Size	2 Maximum size FDDI frames per connection
Frame Size Assembled	85 ATM cells
Peak-to-Average Ratio	2
Burst Factor	1000,500,250

Table 3: Input Parameters : Vary Burst Factor

Table 3 lists parameter settings for the experiment. There are eight active connections, each of which is assumed to have a frame size (negotiated at set up) equal to the maximum size of a FDDI frame. The offered load from the ATM network is increased in steps, while all other hosts generate 1 Mbps background traffic.

Figure 13 shows the variations in gateway throughput as the offered load (average ATM bandwidth) changes. This figure indicates that upto a threshold, gateway throughput increases linearly with the load and is independent of the burst factor of the source. Beyond this threshold however, the throughput begins to level off. The reason for this is that the throughput is restricted to the FDDI link speed of 100 Mbps. At higher loads, the throughput is lower for higher burst factors. This is because of increased cell and frame loss at high loads. An ATM cell which has an overall size of 53 bytes, contains 8 bytes of overhead information and 45 bytes of data. Gateway hardware strips off the 8 bytes of overhead information and uses it to reassemble the 45 bytes of data into a MCHIP segment. This explains why there is approximately a 15 percent difference between the offered load and the carried load in the region below the threshold bandwidth, although there is no cell/frame loss in this region.

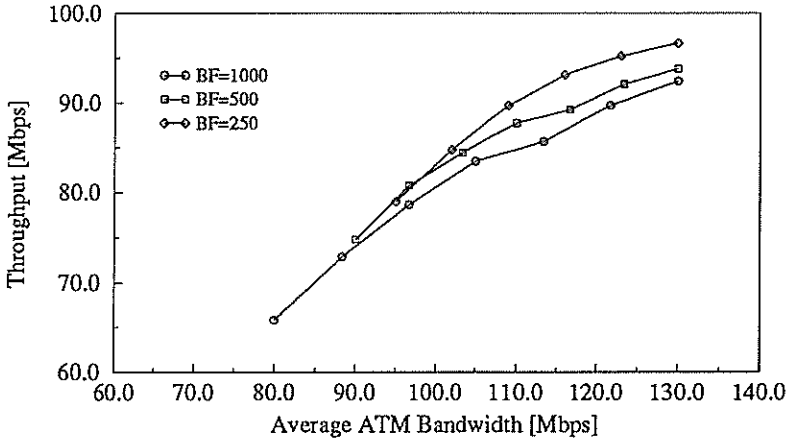

Figure 13: Offered Load vs. Throughput

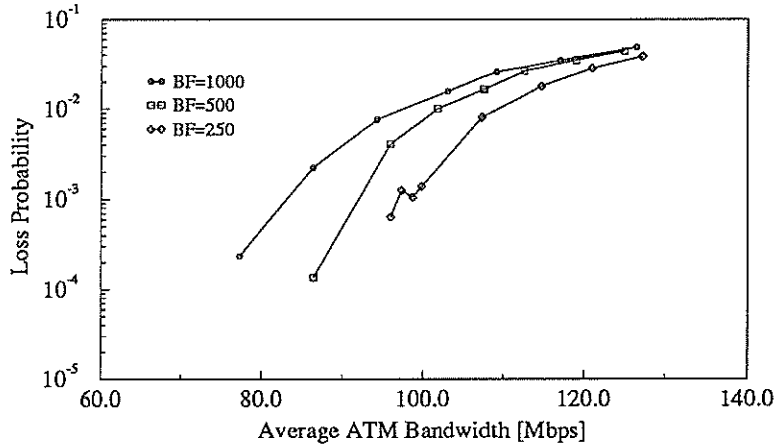

Figure 14: Cell Loss Probability vs. Offered Load

Figure 14 shows the variation of cell loss probability in the SPP with the offered load, at various burst factors. The cutoff bandwidth beyond which the SPP begins to drop cells decreases for smaller burst factors. Moreover, the cutoff bandwidth appears to be sensitive to the burst factor. At loads close to the cutoff bandwidth, the loss probability is small. However at very high loads, loss probability can be as high as 5 percent.

Figure 15 shows how frame loss probability varies with a changing load. Note that the cutoff bandwidth for the frame loss probability is the same as that obtained from the cell loss probability plots. Even a small cell loss probability in the SPP translates to a large frame loss probability in the MPP. Intuitively, this is correct, since even dropping a single cell in the segment being assembled results in the rest of the segment being discarded. This phenomenon becomes especially relevant, when large segments are reassembled under heavy load.

Figure 16 shows the variation in gateway latency for completely assembled frames with changes in the burst factor and the load. At low loads, the gateway latency is dominated by reassembly delay, which decreases exponentially with a fixed number of active connections and increased offered load. Observe that at very high

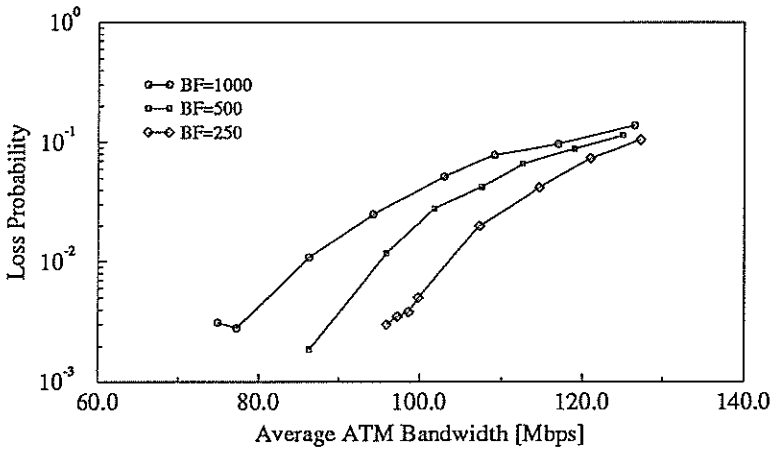

Figure 15: Frame Loss Probability vs. Offered Load

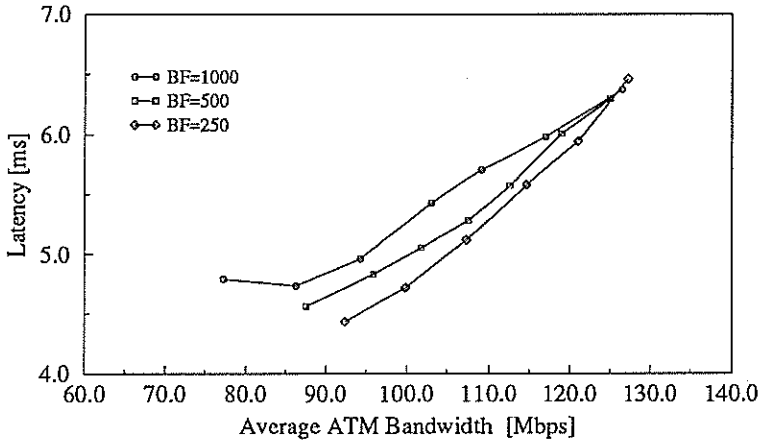

Figure 16: Gateway Latency vs. Offered Load

loads, the latency experienced by a frame is independent of the burst factor. One of the reasons for this is that at high loads, the transmit buffer occupancy reaches its maximum, and incoming frames always see the transmit buffer as being full. Another reason is that the service mechanism for the transmit buffer queue is the periodic timed-token-rotation mechanism in FDDI, and so the average delay experienced by the frame when the transmit buffer is almost full is independent of the input traffic characteristics.

6.2 Effect of Peak-to-Average Ratio of the Source

The peak-to-average bandwidth ratio of a source is another measure of the burstiness of a source, since it effects the rate at which the source generates traffic when in the active state. A large peak-to-average ratio would make the source generate a large burst of traffic in a short time interval; this would mean that large buffers could be requested at very short notice.

In our experiments, the burst factor of the source was 1000. The transmit buffer size, the reassembly buffer

size and the segment size are the same as in Table 3. All the experiments measure various performance metrics as the offered load is varied, for peak-to-average ratios of 2, 4 and 8. Note that at a peak-to-average ratio of 8, when the offered average load is 100 Mbps, the peak offered load is 800 Mbps, which is well over the FDDI link bandwidth of 100 Mbps. For a transient period this can strain the gateway hardware significantly. Such high peak-to-average ratios will be be common in high bandwidth applications of the future [Cox, 1991].

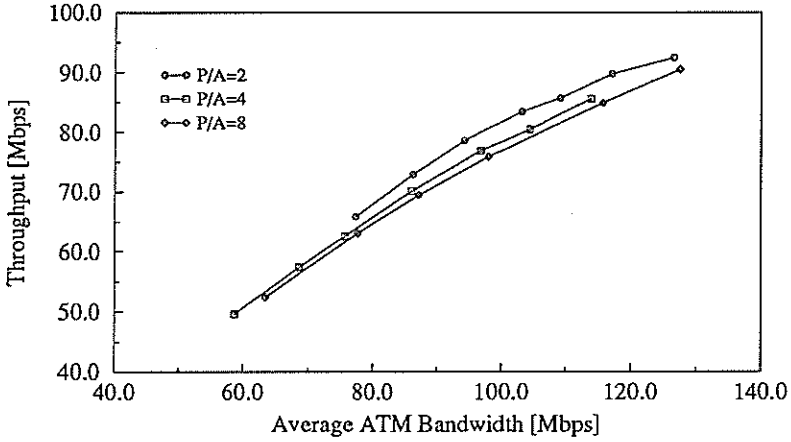

Figure 17: Throughput vs. Offered Load

The gateway throughput performance for different peak-to-average ratios is plotted in Figure 17. Observe that for high loads, the throughput is lower for higher peak-to-average ratios dueto the increased frame and cell loss.

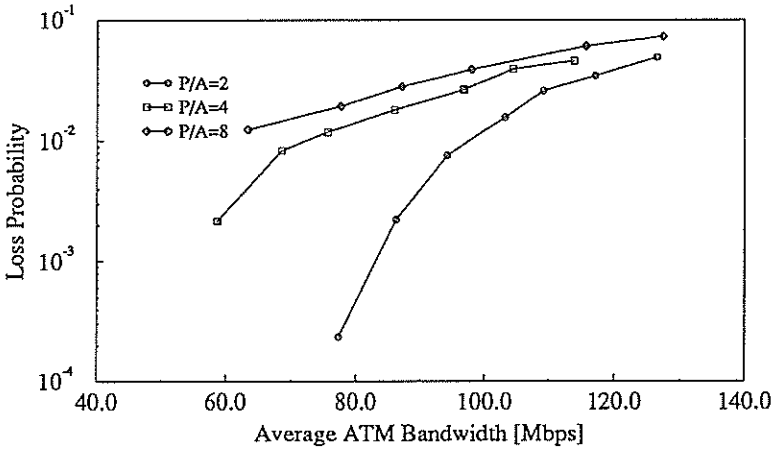

Figure 18: Cell Loss Probability vs. Offered Load

The dependence of cell loss probability on the offered load for different peak-to-average ratios is shown in Figure 18. At high loads, doubling the peak-to-average ratio increases the cell loss probability dramatically. For example, with an offered load of 120 Mbps and a peak-to-average ratio of 2, the source generates traffic at a peak rate of 240 Mbps. If the peak-to-average ratio is doubled, the peak rate is increased to 480 Mbps, which is around

five times the FDDI link capacity. Therefore, with a high peak-to-average ratio, the gateway hardware is forced to assemble a continuous train of back-to-back cells. This causes buffer overflow resulting in significant cell loss.

6.3 Effect of the Node Processing Delay

Earlier, we presented the queueing model for the forward path in the gateway hardware. This model shows that the reassembly buffer is serviced by a complex service mechanism that involves the MPP and NPE, and depends on the transmit buffer occupancy. If the reassembly buffer is not drained at a fast enough rate, the SPP will begin to drop cells. An example of a scenario that could result in such a situation is when the NPE is slow and the gateway has to assemble small frames to meet the demands of a high bandwidth application. In order to study the behavior of the gateway under these conditions, two sets of experiments were conducted. In the first set of experiments, a small burst factor (= 4) was used, while the second set used a moderate burst factor (= 43). In either case, the size of the frame being assembled was small. Both of these experiments were run with the transmit buffer and reassembly buffer size the same as in Table 3. The segment size was fixed at 8 cells, and a peak-to-average ratio of 2 was used. The NPE was configured so that requests were processed at a rate R, varied from 25 to 100 requests per microsecond.

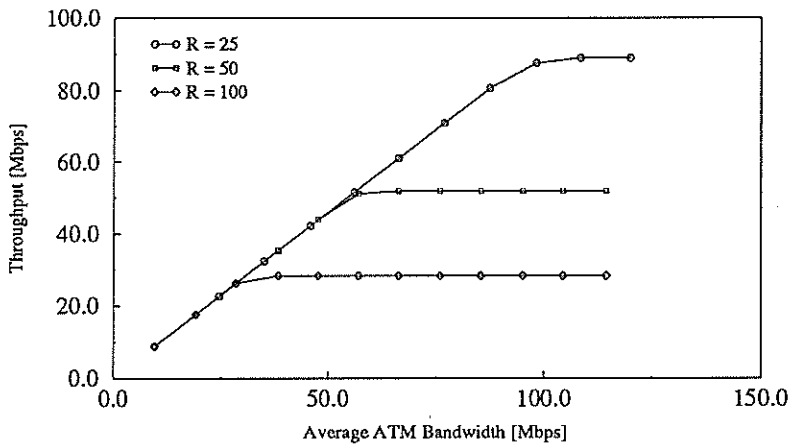

Figure 19: Throughput vs. Offered Load

Figure 19 is a plot of the gateway throughput versus the offered load for different NPE processing rates. It is seen that for a slow NPE, the maximum achievable throughput is restricted by the NPE rate. Any offered load greater than NPE rate results in heavy cell and frame loss. In other words, given the NPE processing rate and the sizes for the receive, transmit, and reassembly buffers, there is a critical minimum segment size. If the segment size drops below this critical minimum, it could result in very high cell/frame loss. Observe also that at low loads, the cells are generated at a slower rate, resulting in a lower request rate to access the transmit buffer. The throughput linearly increases upto a "knee" point at which the request rate just matches the rate at which the NPE can

process the requests. At all loads above this point, throughput remains constant and equals the maximum rate at which segment write requests from the MPP to the NPE can be handled. For loads above the knee, the NPE becomes a bottleneck and the SPP will drop cells. As expected, this will result in increased cell loss probability as the load increases. This can be verified from the Figure 20, which shows the cell loss probability variation for various processing rates and burst factors. It can be seen that the cell loss dramatically increases with the use of a slower node processor and increases exponentially with load, once the throughput saturation sets in.

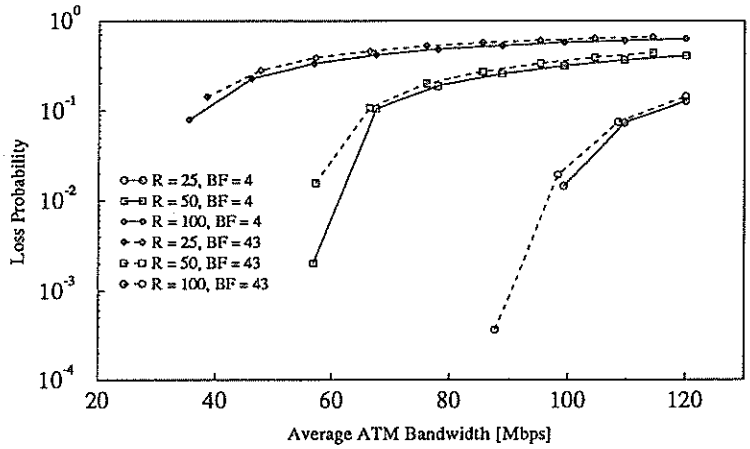

Figure 20: Cell Loss Probability vs. Offered Load

6.4 Effect of Transmit Buffer Size

As long as an application uses large frames and a sufficiently large transmit buffer is available, the reassembly buffer and the NPE can be engineered so that cell/frame loss is kept low. In what follows, we discuss the effects of transmit buffer size on cell/frame loss under very bursty traffic conditions. Note that when the traffic is very bursty and high load conditions exist, the transmit buffer becomes full more often. The MPP will place a hold on acknowledgments sent to the SPP until space is created in the transmit buffer by successful transmissions over the network. The longer the hold on the acknowledgements, higher the probability that the allocated reassembly buffer fills up and the SPP is forced to drop the incoming cells.

The transmit buffer size was set to 16, 32 and 68 full size FDDI frames during the course of the experiments, and the effect of each of these sizes on the performance metrics was observed. The sources used were extremely bursty, with a peak-to-average ratio of 8 and a burst factor of 1000. The reassembly buffer size was the same as in Table 3.

In Figure 21, we see the effect of variations in transmit buffer size on the throughput versus offered load curve. As expected, a larger number of buffers improves the gateway throughput significantly.

Figure 22 plots the cell loss probability as a function of offered load for different transmit buffer sizes. The cutoff

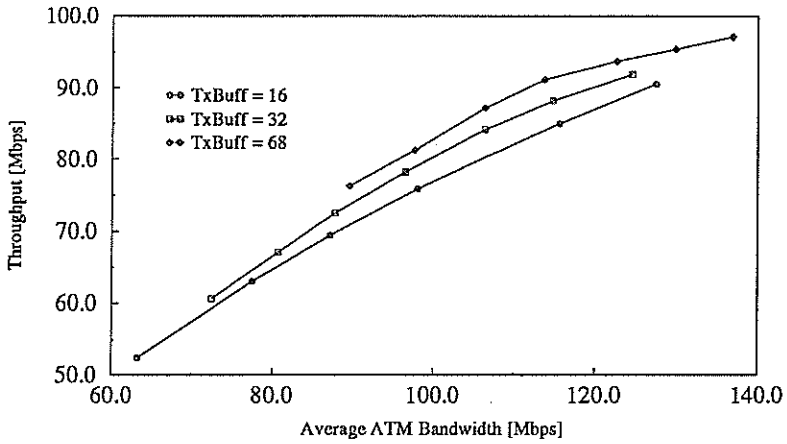

Figure 21: Gateway Throughput vs. Offered Load for various Transmit Buffer Sizes

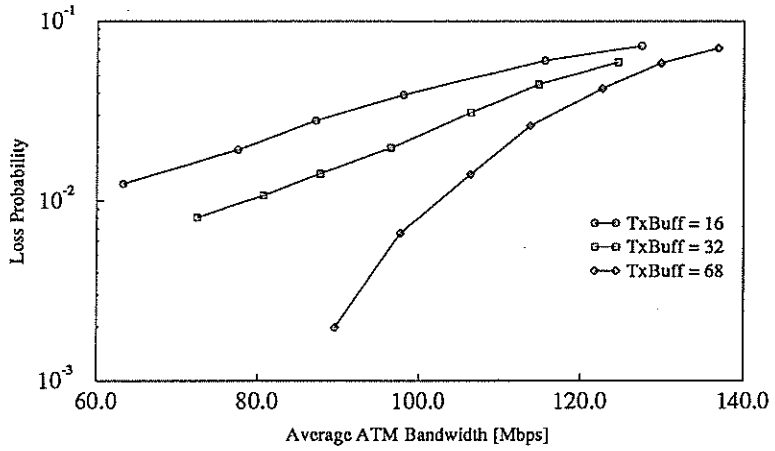

Figure 22: Cell Loss Probability vs. Offered Load

bandwidth strongly depends on the transmit buffer size. It is seen from the figure that the maximum transmit buffer size allowed by AMD's SUPERNET chip set is inadequate if we wish to support a high load that exhibits a high degree of burstiness. Also, note that since the other hosts on the FDDI network are generating a minimal amount of traffic, the gateway FDDI interface does not have to wait for a token. If the FDDI network is heavily loaded and the gateway is receiving bursty traffic, the observed loss can be even higher.

Figure 23 shows how the latency versus offered load curve behaves when subjected to different transmit buffer sizes. Clearly, larger transmit buffer sizes result in greater delay through the gateway. With the increase in the offered load, the gateway latency increases exponentially. When the transmit buffer size is set to its maximum allowable value, the gateway latency is observed to be remarkably higher at larger values of the offered load. This increased delay is caused by the fact that there is more queueing at the transmit buffer. Therefore, for very bursty traffic, there is a tradeoff between cell/frame loss and gateway latency; this tradeoff can be made to tilt one way or another by adjusting the size of the transmit buffer memory.

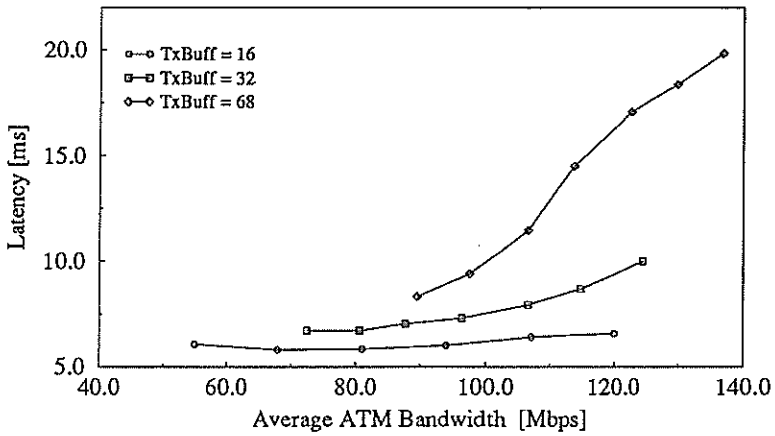

Figure 23: Gateway Latency vs. Offered Load

6.5 Experiments with the FDDI to ATM Path

In Figure 4 we showed a simplified queueing model for the reverse data path. It can be seen that the service mechanisms in this queueing system are simpler than their counter parts in the queueing model for the forward pipeline. If the receive buffer has sufficient space, the reverse data path should be able to handle peak 100 Mbps traffic from FDDI. We carried out an experiment (which has not been reported here), which proves that a receive buffer equal to 16 full size FDDI frames is sufficient for very bursty traffic on FDDI destined to an ATM host.

7 Conclusions

This paper presented a detailed simulation model for the design of a high performance ATM-FDDI gateway based on a new connection- oriented internetwork abstraction called VHSI. The simulation model, constructed using a graphical simulation tool called BONES, served the dual purpose of functional verification and performance evaluation.

Under typical traffic parameters such as burst factors less than 1000, peak-to-average ratio of 2 and a small transmit buffer size that represent common case processing, a reassembly buffer with buffer space equivalent to two maximum sized FDDI frames per connection is sufficient to obtain acceptable cell loss in the range of 10^{-4}. With full size transmit buffers, the cell loss behavior is significantly improved and gateway can deliver almost ideal throughput.

The experiments that study the performance for the input traffic characteristics indicate that high burst factors (≥ 1000) and high peak to average ratio (≥ 8) for bursty applications can result in intolerable cell loss ($\approx 10^{-2}$) and frame loss ($\approx 10^{-1}$), if the transmit buffer is not adequate. Also the cutoff bandwidth above which the gateway drops cells and frames is very sensitive to the input traffic.

The experiments that study the effect of the transmit buffer size indicate that under very bursty traffic conditions, the maximum allowable transmit buffer size of 256 KB (68 full size FDDI frames) is not sufficient for obtaining zero cell loss in the gateway. At high loads and with adequately fast NPE, restricted transmit buffer capacity can cause significant cell loss and loss in throughput.

It has also be seen that the architecture of the gateway is constrained by restrictions imposed by the design of AMD's FDDI interface chip set. It requires that the access to the transmit buffer should be obtained through the intervening Node Processing Element (NPE), which should be sufficiently fast, otherwise if the gateway is assembling small sized frames at high loads, the NPE can become a bottleneck. This fact was conclusively proved by the experiments describing the effect of the node processing elements.

Also, the maximum achievable gateway throughput is restricted by the maximum rate at which requests of MPP to access the transmit buffer can be handled. Note that no resource management and enforcement functions were modeled and hence no control traffic was simulated. It was assumed that the control packet always receive the prioritized service needed from the NPE. In light of this fact engineering the NPE adequately is very important.

It was also found that due to simpler processing needs of the FDDI to ATM path, there are no bottlenecks in this path and the gateway can successfully carry 100 Mbps load from FDDI to ATM side.

This paper also highlights the utility of a high level graphical simulation tool such as BONES in developing complex simulation models.

8 Acknowledgements

The authors would like to thank Zubin Dittia for carefully reading the draft and simplifying the text to improve the readability. The authors would also like to thank Lorrie Ackerman for reading the paper and pointing out some peculiarities of technical writing.

References

[AMD, 1990] Advanced Micro Devices Inc. (1990) *The SUPERNET Family for FDDI.*

[ANSI, 1987] American National Standards Institute Inc. (1987) ANSI X3.139-1987 *Fiber Distributed Data Interface (FDDI), Token Ring Media Access Control (MAC).*

[Anderson, 1991] Anderson, J. (1991) 'Persistent Connections in High Speed Internets', MS thesis, Department of Computer Science, Washington University in St. Louis, Missouri.

[Anderson *et al.*, 1991] Anderson, J., Parulkar, G.M., and Dittia, Z. (1991) 'Persistent Connections in High Speed Internets' *Proceedings of the IEEE GLOBECOMM'91.*

[ANSI, 1990] ANSI T1S1 Technical Sub-Committee. (1990) 'Broadband Aspects of ISDN Baseline Document,' T1S1.5/90-001.

[Cox, 1991] Cox, J.R.Jr., Turner, J. (1991) 'Project Zeus: Design of a Broadband Network and its Application on a University Campus' Technical Report WUCS 91-45 , Department of Computer Science, Washington University in St. Louis, Missouri.

[Dehart *et al.*, 1991] Bubenik, R.G., DeHart, J.D., Gaddis, M. (1991) '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*, 59-68.

[CCITT, 1991] CCITT, 'Report of Working Party XVIII/8 (General B-ISDN Aspects),' Study Group XVIII–Report R70, July, 11-28.

[Escobar, 1991] Escobar, J., and Partridge, C. (1991) 'A Proposed Segmentation and Reassembly Protocol for Use with Asynchronous Transfer Mode (ATM),' *Protocols for High Speed Networks*, II, (M. Johnson, ed), 353-368.

[Comdisco, 1991a] Comdisco Systems Inc (1991) *BONeS Users Guide.*

[Comdisco, 1991b] Comdisco Systems Inc. (1991) *Block Oriented Network Simulator (BONeS) Additional LAN Models*, 1991.

[Cranor, 1991] Cranor, C. (1991) 'An Implementation Model for Connection-oriented Internet Procols,' MS thesis, Department of Computer Science, Washington University in St. Louis, May.

[Gaddis, 1989] Gaddis M. (1989) 'FPP Packet Formats and Link Level Protocol Descriptions,' *Working Note ARL-89-07*, Department of Computer Science, Washington University in St. Louis.

[Kapoor, 1991a] Kapoor, S., Buddhikot, M., M. Parulkar G.,M. (1991) 'Design of a Two Port ATM-FDDI Gateway,' to appear in *Journal of Internetworking : Research and Experience*

[Kapoor, 1991b] Kapoor, S. (1991) 'Design and Analysis of a Two Port ATM-FDDI Gateway' MS thesis, Department of Electrical Engineering, Washington University in St.Louis, December.

[Katz, 1992] Katz D. (1992) 'Transmission of IP and ARP over FDDI Networks' *Internet Engineering Task Force, Internet Draft.*

[Kent, 1987] Kent C.A and Mogul J.C. (1987) 'Fragmentation Considered Harmful,' *Proceedings of the ACM SIGCOMM'87*, 390-401.

[Mazraani, 1990a] Mazraani, T., Parulkar, G.M. (1990) 'Specification of a Multipoint Congram-oriented High Performance Internet Protocol,' *Proceedings of the IEEE INFOCOM'90,* 450-457.

[Mazraani, 1990b] Mazraani, T. (1990) 'High speed internet protocols and resource management in the ethernet,' MS thesis, Department of Electrical Engineering, Washington University in St. Louis, May.

[Parulkar, 1990] Parulkar, G.M. (1990) 'The Next Generation of Internetworking,' *ACM SIGCOMM Communication Review*, **20**, (1), 18-43.

[Turner, 1988] Turner, J. (1988) 'Design of a Broadcast Packet Switching Network,' *IEEE Transactions on Communications*, **36**, (6), 734-743.