PCnet-*FAST* Buffer Performance White Paper

The PCnet-FAST controller is designed with a flexible FIFO-SRAM buffer architecture to handle traffic in half-duplex and full-duplex 100-Mbps Ethernet networks. This buffer architecture provides high performance by keeping overflows and underflows to a minimum for various system and network environments.

In window-based protocol schemes like TCP and IPX, overflows and underflows are particularly expensive since the missed frame involves retransmission of that particular frame (or more frames within the window), resulting in a reduced overall network and system throughput. The following study indicates a need for large buffers to avoid costly underflows and overflows in these environments.

APPROACH

A system-level model of the PCnet-*FAST* buffer architecture was developed using a modeling tool called Workbench from SES [ref. 1]. Workbench provides the ability to model complex queuing systems in a detailed fashion. System functions are abstracted to queues, delays, interrupts, resources, and transactions, each of which represents the activities that systems typically process.

For modeling purposes, the PCnet-*FAST* controller is divided into three distinct interfaces as shown in Figure 1. The network interface models the reception and transmission of frames over the wire. Almost all the operational parameters in this interface are defined in the IEEE 802.3 standard [ref. 2]. Two important network traffic-related parameters, packet size distributions and packet arrival rates, were derived based on measurements for a typical client-server configuration.

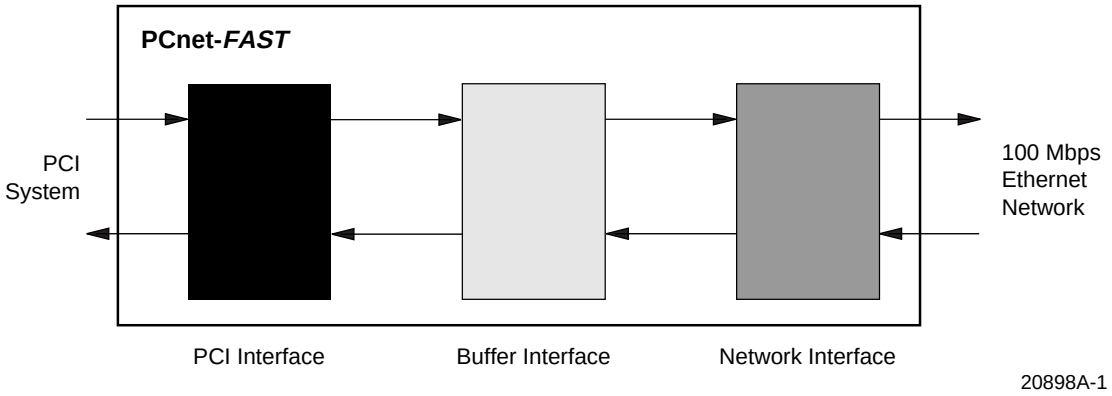


Figure 1. PCnet-*FAST* Model Architecture

The buffer interface includes the FIFO-SRAM subsystem. The timings and operations mimic the actual implementation. Sensitivity analysis was used to identify critical parameters in the buffer interface.

The PCI interface transfers data between the system buffer and the FIFO-SRAM subsystem as defined by a typical PCI bus transaction [ref. 3]. Figure 2 shows one possible PCI system configuration. The chipset acts as a bridge between the host bus and the PCI bus. All host bus-to-PCI bus, CPU-to-system memory, and PCI bus-to-system memory operations go through the chipset. The chipset also provides bus arbitration for the devices residing on the PCI bus.

Publication# **20898** Rev: **A** Amendment/**0**
Issue Date: **June 1996**

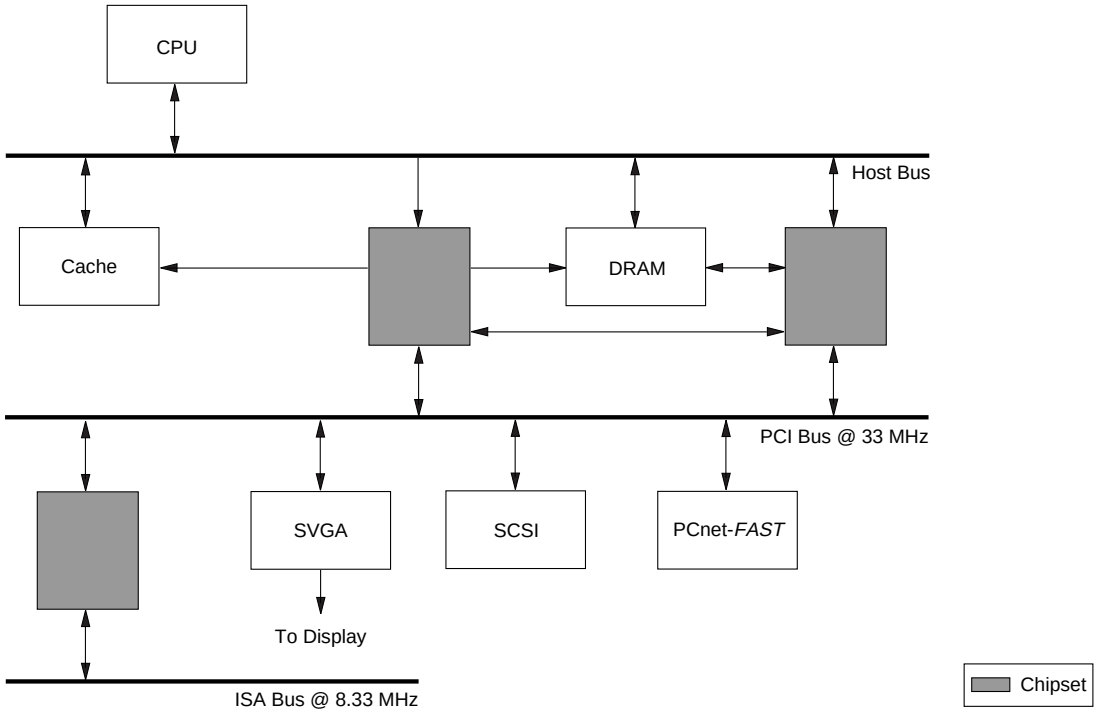


Figure 2. Sample PCI System Configuration

Bus latency and **burst size** are the two most important parameters for the PCI interface model. Bus latency defines the interval from the time a PCI device requests the bus to the time it can begin the first data transfer. Burst size is the number of bytes that can be transferred by a master each time it gains control of the PCI bus. The bus latency and burst size together determine the ability of the system to drain the data out of the network controller buffer. Smaller burst sizes and longer latencies mean longer service times and larger controller buffer requirements. The burst sizes and bus latencies vary widely from system to system, depending on configurations and applications. While some of the recent chipsets allow transfer upwards of 64-bytes of data each time the bus master has an access to the PCI bus, most other chipsets do not burst more than 16-bytes[1]. While conducting performance studies of various PC systems, it was found that PCI bus latencies can be as high as 6–8 µs for a client and as high as 10 µs for a server. The higher bus latencies can be seen in systems with ISA controllers and older chipsets.

Underflows and overflows are important in determining the network controller buffer requirements. An underflow can occur when the transmit buffer receives data from the system at a rate slower than the network rate. Overflow occurs when a system cannot drain out the data from the receive buffer at a sufficient rate, causing the buffer to fill up, and the rest of the data on the wire to be thrown away. Underflows and overflows in any environment are costly as they involve retransmission by the sender, causing a reduction in the effective system and network throughputs. Let's examine this from the point of view of two common protocols: **TCP/IP and IPX**.

Both TCP and the latest versions of IPX use a windowing protocol scheme for data transmission. Instead of sending one frame and waiting for the acknowledgment from the receiver before transmitting the next frame, the protocol allows for sending a burst of frames by the transmitter before receiving an acknowledgment from the receiver. The amount of burst is determined by the size of the window. It has been reported that increasing the window size greatly improves the overall network performance [refs. 4, 5]. If the sender does not receive an ACK within a predefined window, some form of retransmission has to occur. Besides network infrastructure problems, the most probable cause for an absence of ACK is the dropping of a data frame by the receiver! Frames can be dropped by the receiver if it does not have enough buffer space.

1. This study is performed for the chipsets used with fifth-generation CPUs, with a caveat that the buffer requirement may be more severe if older chipsets are used.

TCP uses the sliding window protocol [ref. 4]. As the ACK for the leftmost frame in the window is received, the window is moved to the right. If the sender reaches the end of the window before receiving an ACK for the leftmost frame, retransmission of the frame missing an ACK is required. Some implementations handle this by retransmitting the whole window, while others may only retransmit the missed frame. In any case, the overhead is not trivial.

VLM and BNETX implementations of IPX [ref. 5] also use the sliding window protocol. However, in these cases, the retransmission can have the effect of reducing the window size for future transmissions. And as mentioned earlier, the reduction in window size has a negative effect on the network performance.

In TCP, the maximum window size can be up to 32 Kbytes and sometimes even higher. IPX defines 16 frame windows for read and 10 frame windows for write (assuming 1500-byte frames, the window size for read is 24 Kbytes). Given these large window sizes for both protocols, any retransmission due to a dropped frame can impact the overall network performance significantly.

With the understanding that overflows and underflows can affect the system and network performance, this study was performed to determine the buffer requirements to ensure zero underflows and overflows under various environments for a typical network controller card.

SIMULATION CASES AND PARAMETERS

For the transmit section to prevent any underflows, the buffer need not be much larger than the maximum packet size (maximum packet size in IEEE 802.3-based networks is 1518 bytes). A transmit buffer of 2–4 Kbytes is sufficient for most applications. Hence, no simulations are run to determine the range of buffer requirements for the transmit side.

For the receive section, the situation is quite different. The network controller buffer requirements vary widely, depending on the type of network and system environments. For a heavily loaded system where the latencies can be high (for example, a PC with multiple PCI and ISA controllers running disk and network intensive applications), network controller buffer requirements need to be large enough to prevent overflows. The following section shows the results of simulation runs for different environments. Various cases covering half- and full-duplex 100-Mbps Ethernet networks are considered, and buffer requirements for the receive section are illustrated in the graphs under each case. The simulation cases presented in this white paper include the following:

Case 1: 100-Mbps Half-duplex for a 24-Kbyte window (IPX window size)

Case 2: 100-Mbps Full-duplex for a 24-Kbyte window.

Case 3: 100-Mbps Full-duplex for a 24-Kbyte window with PCI clock = 25 MHz

Case 4: 100-Mbps Half-duplex for a 32-Kbyte window (TCP window size)

Case 5: 100-Mbps Full-duplex for a 32-Kbyte window

Case 6: 100-Mbps Netbench trace for a client

Case 7: 100-Mbps Netbench trace for a server

Simulation parameters

Wire speed: 100-Mbps

Inter Frame Space (IFS): 10 μs[2]

PCI Clock: 33 MHz (unless stated otherwise)

PCI Bus cycle: First data phase - 7 PCI clocks; subsequent data phases - 1 PCI clock (7-1-1-1. cycle)

Burst Size: 64, 80, 96-bytes

Simulations are carried out for no collisions on the wire.

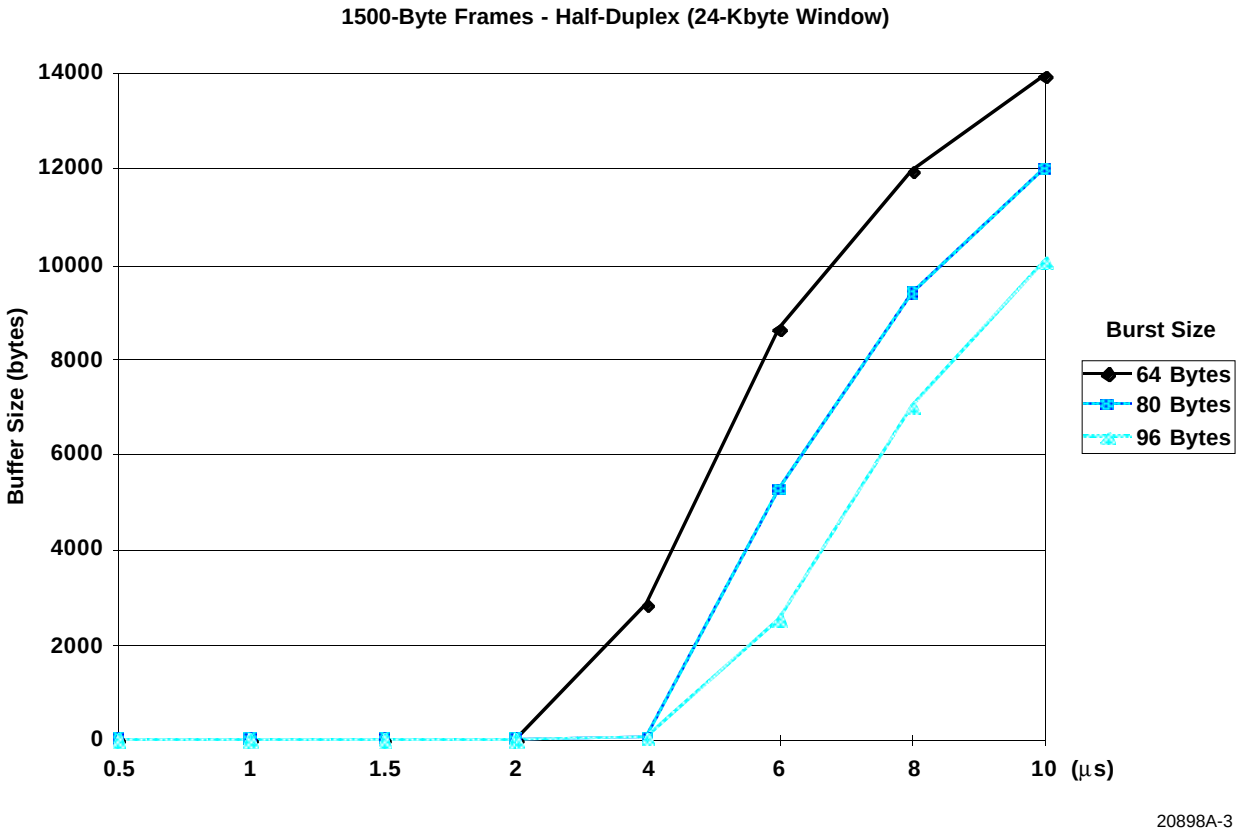
[2] *If a more aggressive IFS is used, the required buffer size will be larger and all other parameters will be the same.*

RESULTS AND ANALYSIS

Graphs are plotted with PCI bus acquisition latencies on X-axis and buffer requirements of PCnet-*FAST* on Y-axis for various PCI burst sizes. Note that the buffer requirements presented here are for the receive section only.

Case 1: 100-Mbps Half-Duplex for a 24-Kbyte Window

The latest versions of IPX(VLM) uses a sliding window protocol with window size of 16 frames for read and 10 frames for write. Assuming a frame size of 1500-bytes, these window sizes in terms of bytes are 24 Kbytes for read and 15 Kbytes for write. To find the buffer requirements for the receive section, the 24-Kbyte window is considered. The resulting buffer requirement is as shown below.



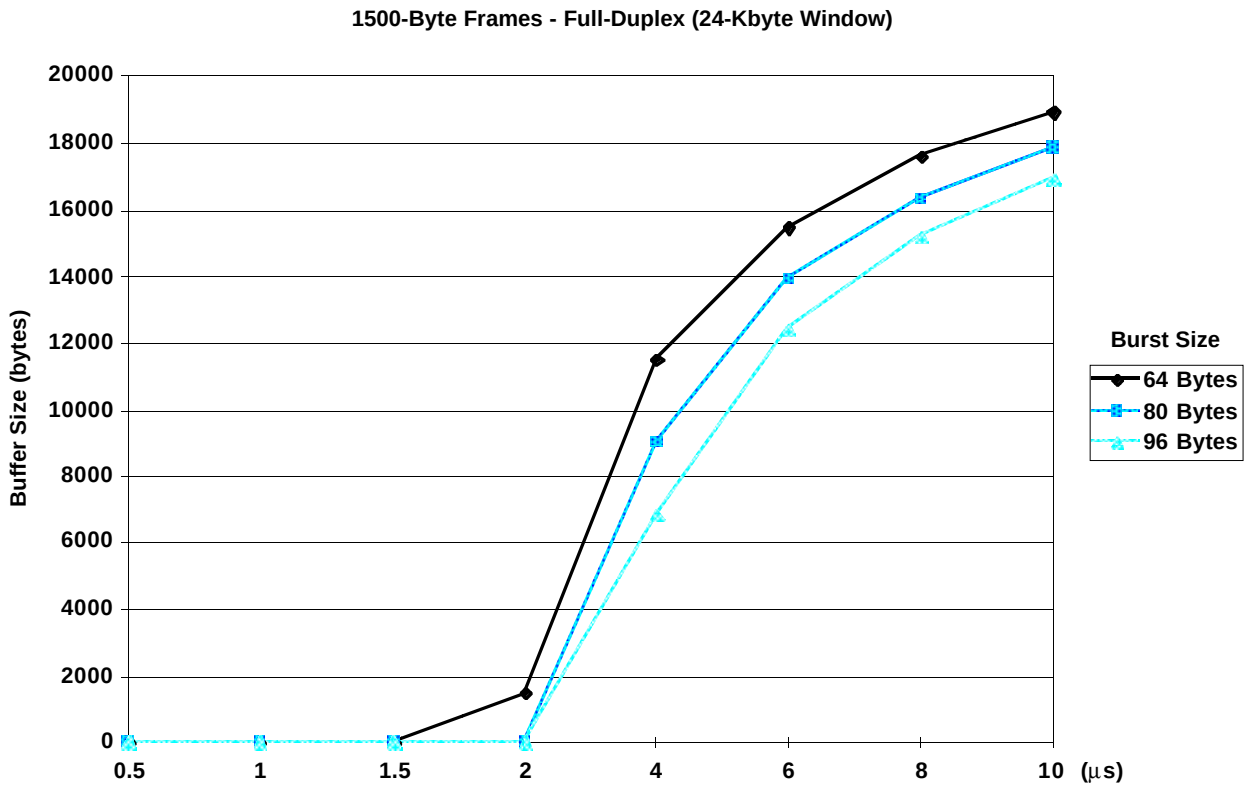
20898A-3

Case 2: 100-Mbps Full-Duplex for a 24-Kbyte Window

The IPX/VLM type traffic is simulated here for a 100-Mbps full-duplex operation.

The results presented in Case 1 and Case 2 give an estimate of the buffer requirements for PCnet-*FAST* operating in a 100-Mbps Ethernet network under IPX environment. As can be seen, for the low bus acquisition latencies, the buffer requirement is minimal. However, as bus latency increases beyond 2 µs, buffer requirements are non-trivial. Providing an SRAM of the size indicated in the graphs guarantees zero overflows for 16 back-to-back frames of 1500-bytes, which is typical IPX/VLM traffic.

The dependence of buffer requirements on burst sizes and acquisition latencies is obvious in the above graphs. Depending on the system configuration, buffer requirements will vary widely. To avoid boundary cases, it is advisable to design for the worst case, especially while designing network adapter cards that may be installed into the existing older machines.

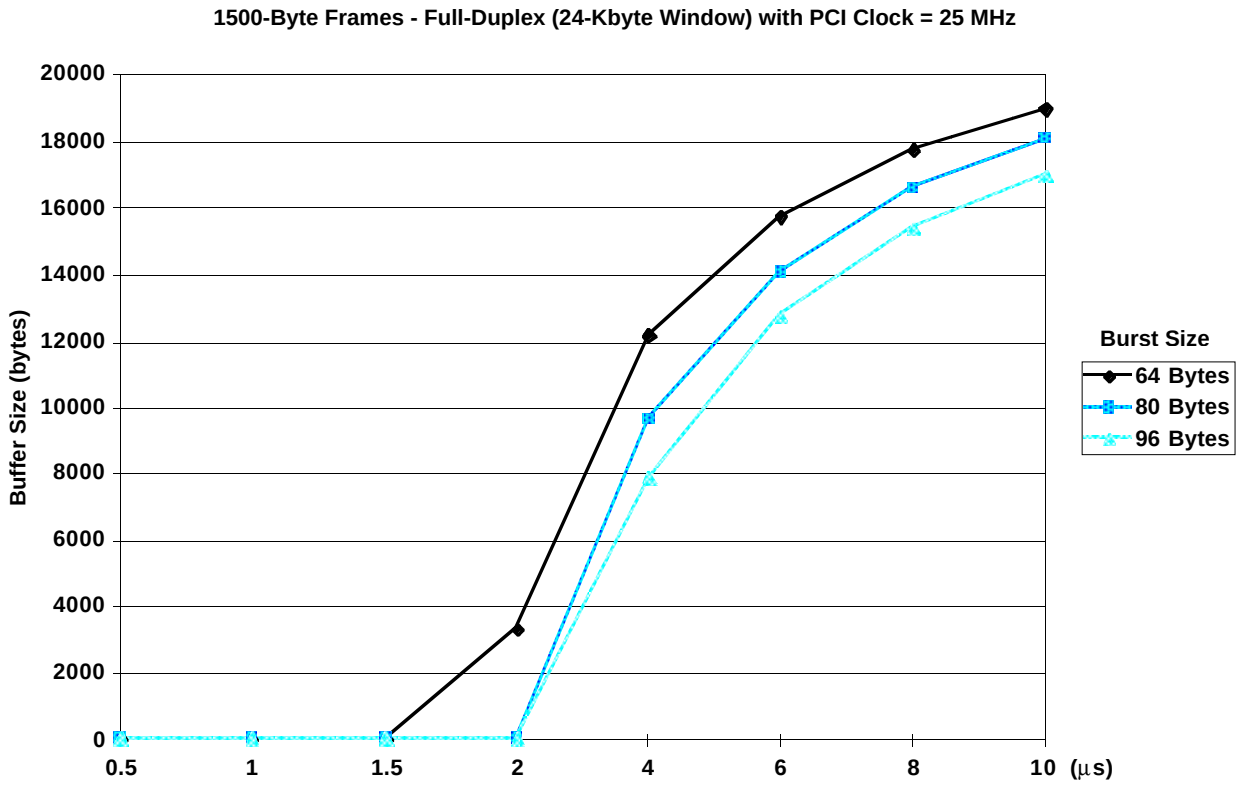


20898A-4

Case 3: 100-Mbps Full-Duplex for a 24-Kbyte Window with PCI Clock = 25 MHz

To understand the effect of PCI clock speed on the buffer requirement, simulation environment of Case 2 was run at 25 MHz instead of the default 33 MHz. The result is as shown below.

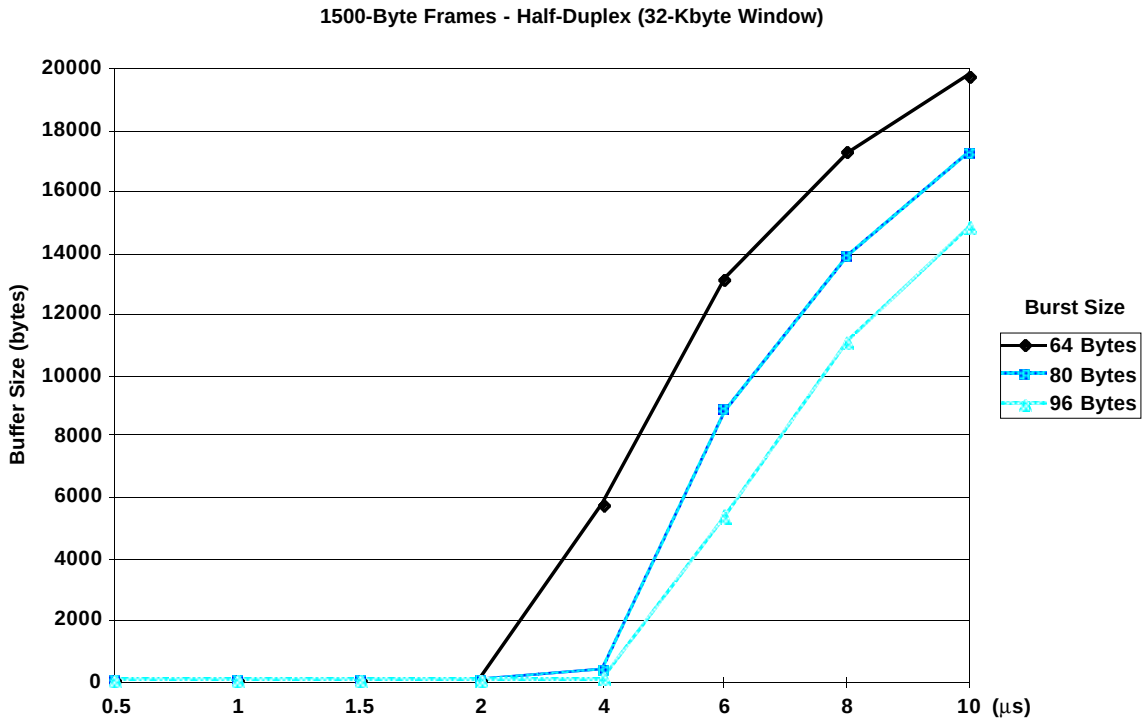
Comparing this graph with the one presented for Case 2, the PCI clock speed is found to be a factor at lower latencies and affects the buffer requirements. However, at latencies beyond 4 µs, there is a minor (if any) increase in the buffer requirements. This small variation with respect to PCI clock speed is expected as the time required to transfer data on the PCI bus is much smaller than PCI bus acquisition latency.



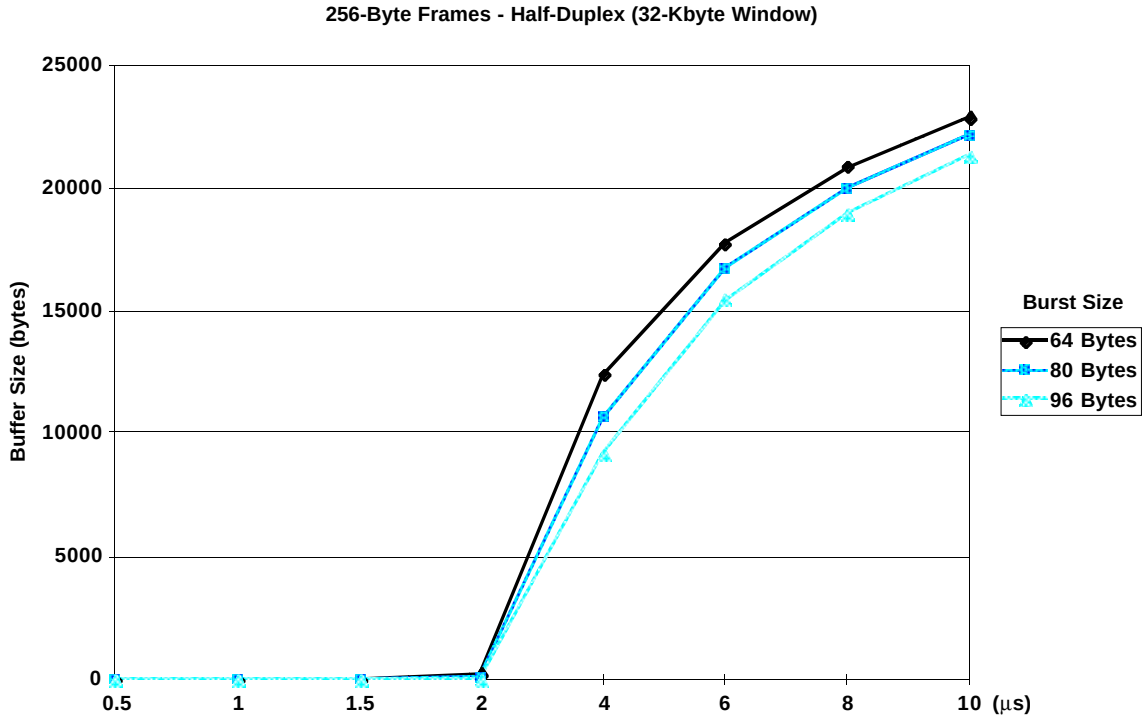
20898A-5

Case 4: 100-Mbps Half-Duplex for a 32-Kbyte Window

TCP, like IPX, uses the sliding window protocol. The window size in the TCP case can be as high as 32 Kbytes. The simulations in the TCP case are carried out for 1500-byte and 256-byte frames. The resulting buffer requirements are as shown below.



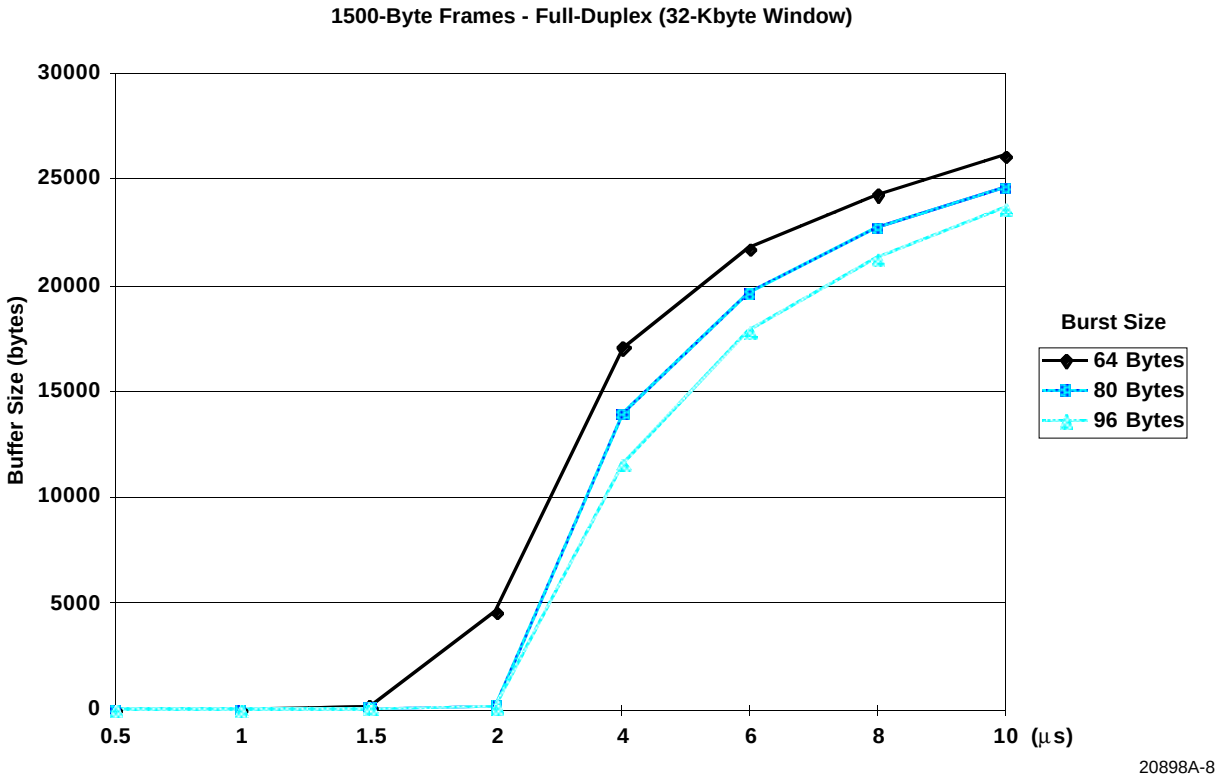
20898A-6



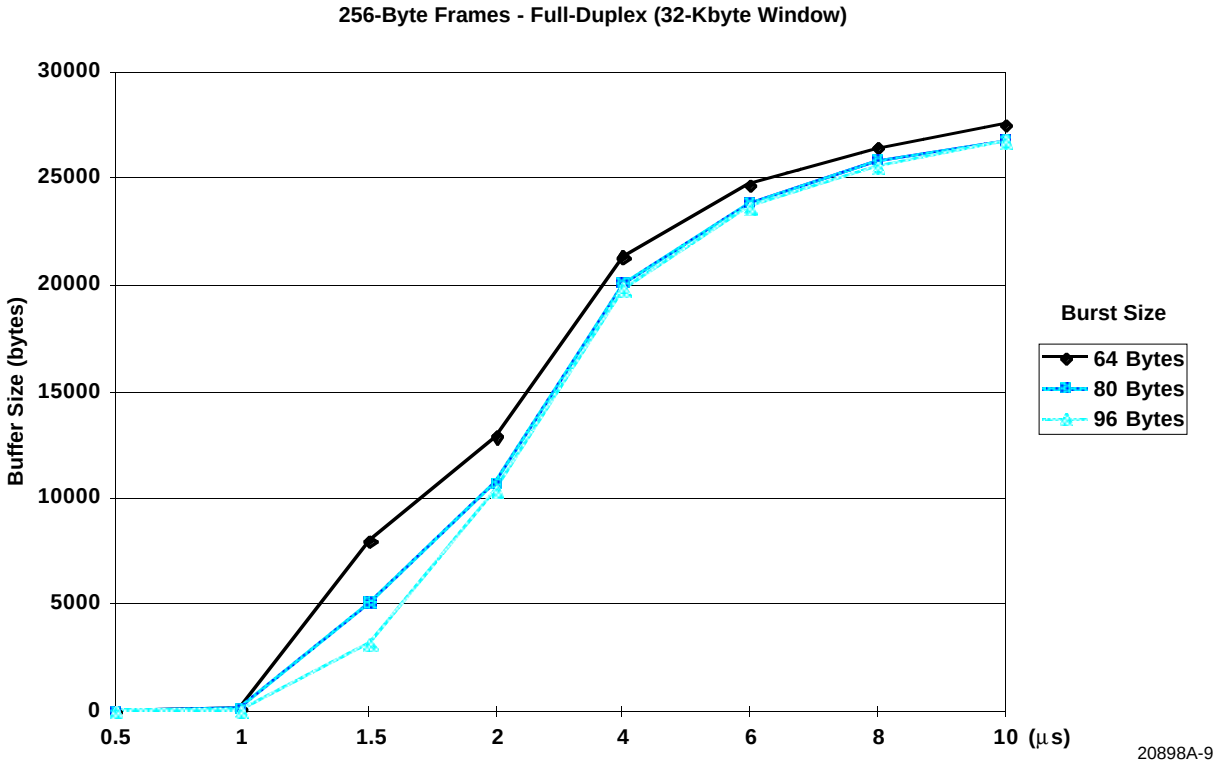
20898A-7

Case 5: 100-Mbps Full-Duplex for a 32-Kbyte Window

The TCP traffic is simulated here for a 100-Mbps full-duplex operation.



20898A-8



20898A-9

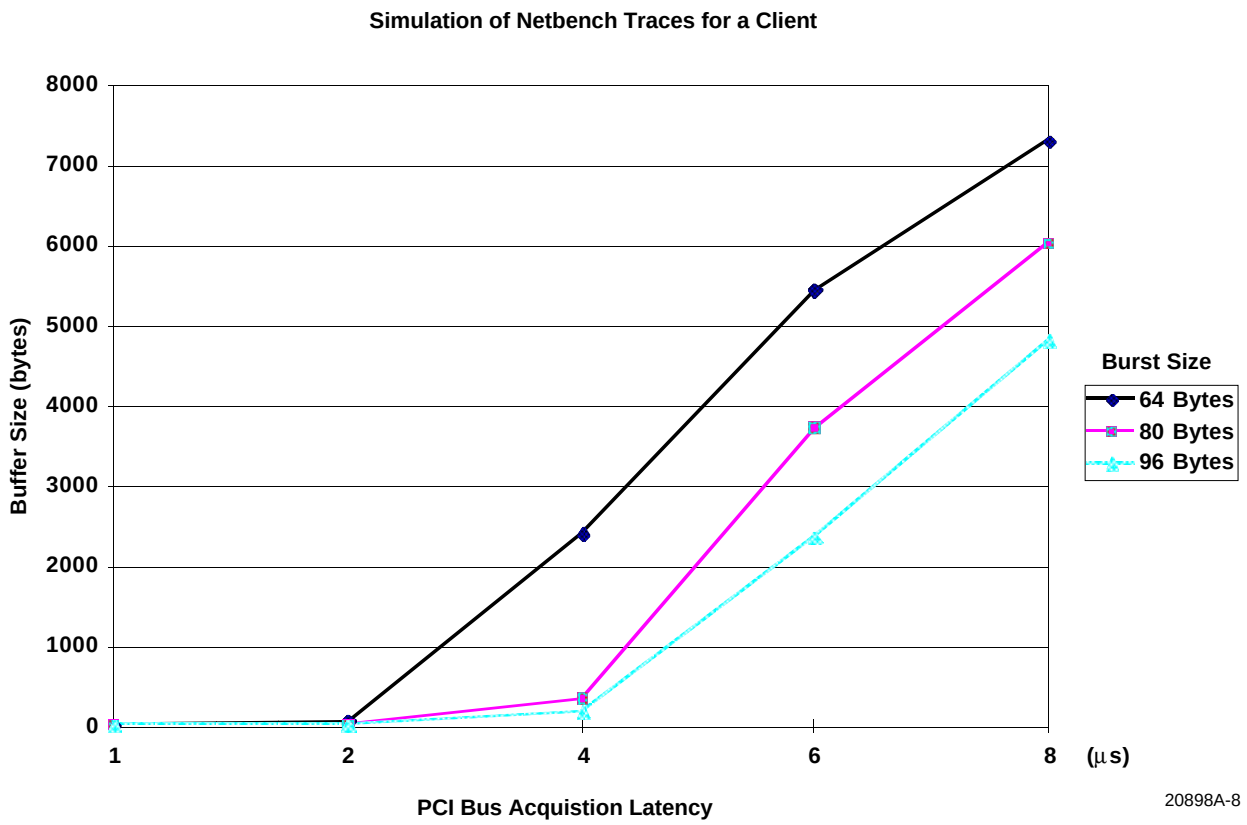
The results presented in Case 4 and Case 5 give an estimate of the buffer requirements for PCnet-*FAST* operating in a 100-Mbps Ethernet/TCP network. Providing a buffer the size indicated in the graphs ensures zero overflows for 32-Kbyte TCP windows.

The buffer requirement for 256-byte frames is larger than for 1500-byte frames, corresponding to the extra overhead associated with processing each received packet.

The analysis for PCI clock = 25 MHz in the TCP case is similar to IPX. Refer to Case 3 for more details.

Case 6: 100-Mbps Netbench Trace for a Client

Netbench 4.0 [ref. 7], which replicates a typical networked PC environment, is used to collect data for the network traffic occurring in a six client/one server configuration. The network traces for the 100-Mbps half-duplex network were captured using a network analyzer from Wandel & Goltermann. These traces were then used as inputs to the simulation environment. The buffer requirement for a client is as shown below.



For this particular trace, the client receives a long burst of almost full-size frames followed by a client think time and transmission of small request frames followed by a second burst of reception.

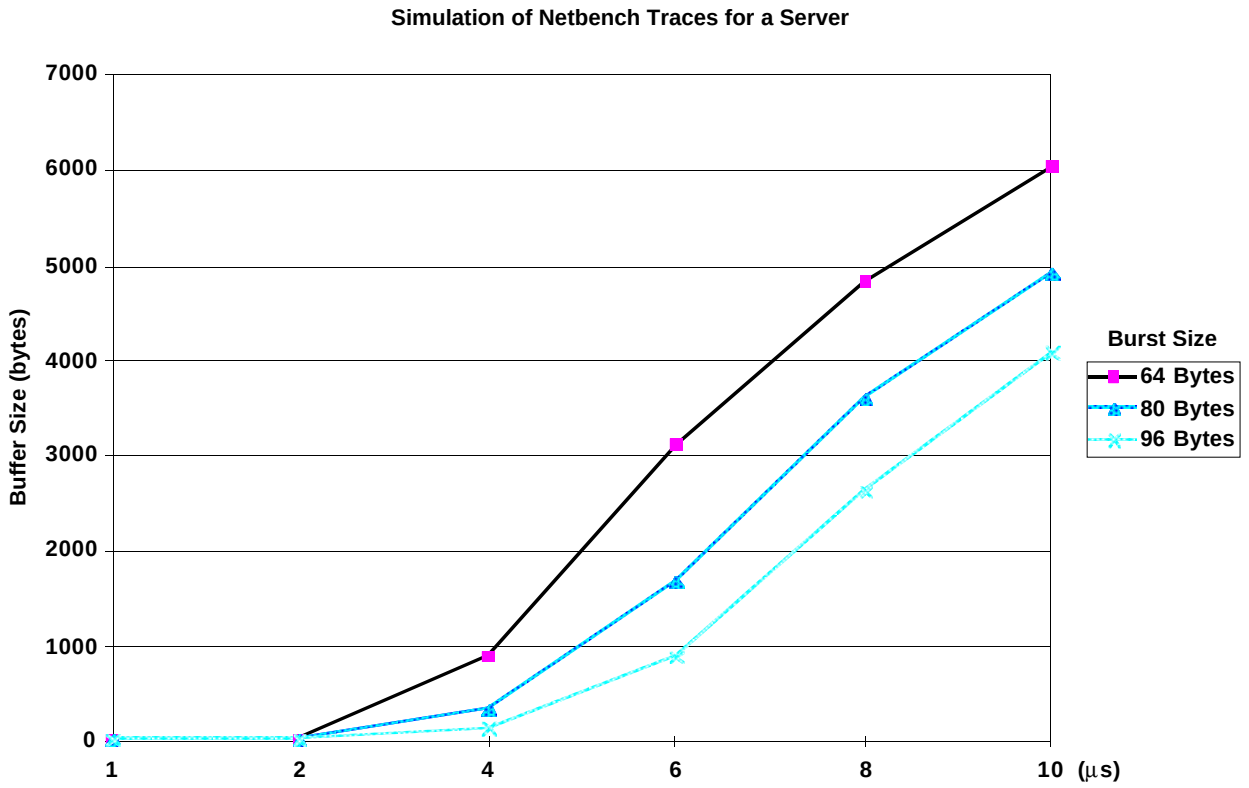
For clients, latencies have been in the range of 6–8 µs. As long as the PCI burst size is 64-bytes or greater, the buffer sizes can be in the range of 5–7.5 Kbytes.

The results presented here are for 100-Mbps half-duplex traffic. Monitoring the network traffic in a full-duplex environment is not feasible using the presently available network analyzers. Hence, no simulations can be run for full-duplex case. However, referring to the Case 2 and Case 5, it can be said that the buffer requirements will be larger compared to the one for a half-duplex network.

Case 7: 100-Mbps Netbench Trace for a Server

The buffer requirements for the receive side of a server are as shown below.

The typical traffic observed in the Netbench setup has the server receiving small frames (requests) followed by a transmission of a stream of long frames (data) to a requesting client. There were a few bursts of long data packets to the server. As the server is mainly involved in the transmission of data, the buffer requirement for the receive section is not as high as for the client. The buffer requirements in a full-duplex operation will be higher compared to the one presented for the half-duplex case. PCI latencies on servers have been as high as 10 µs for certain system configurations.



20898A-9

CONCLUSIONS

Since the system and network environments vary widely from one configuration to another, the data in this white paper is intended to provide an insight into the effects of various parameters on network controller buffer requirements. By identifying the typical parameters in your environment and then using the graphs presented above to determine the buffer requirements, overflows/underflows in a system can be avoided, thereby enhancing the overall system and network throughput.

Based on the data presented in this white paper, the following conclusions can be drawn:

Receive Section

- PCI bus latency and PCI burst size are two important parameters in determining the network controller's buffer requirements.
- For presently available systems, the buffer requirements of a network controller may not be trivial unless low latency can be guaranteed for all times. To avoid costly overflows and to take maximum advantage of large TCP and IPX window sizes, a system should be designed with optimum buffers for that configuration.
- The PCI clock speed has a mitigating effect on the network controller's buffer requirements as the PCI bus acquisition latency increases. At latencies over 4 µs, PCI clock speeds of 25 MHz and 33 MHz have almost the same buffer requirement.
- An external buffer is necessary for the PCnet-*FAST* controller when used at 100-Mbps to generate optimal performance.

Transmit Section

- The transmit buffer need not be much larger than the maximum packet size. A transmit buffer of 2–4 Kbytes is sufficient for most applications.
- In a full-duplex 100-Mbps Ethernet environment with high bus latency, underflows can be prevented by buffering an entire packet in the PCnet-*FAST* controller before the start of transmission.

REFERENCES

1. *SES Workbench User's Manual*, Scientific and Engineering Software Inc.
2. *IEEE 802.3 Standard Specifications*, IEEE Publications
3. *PCI Local Bus Specification* Rev 2.1, PCI Special Interest Group
4. *TCP/IP Illustrated*, Vol. 1, W. Richard Stevens, Addison Wesley
5. *Packet Burst Update: BNETX vs. VLM Implementations*, Novell Research
6. *PCnet-FAST Preliminary Data Sheet*, Advanced Micro Devices
7. *Understanding and Using Netbench 4.0*, Ziff-Davis Publishing

Trademarks

AMD, the AMD logo, and combinations thereof are trademarks of Advanced Micro Devices, Inc.

Am186, Am386, Am486, Am29000, *b*IMR, eIMR, eIMR+, GigaPHY, HIMIB, ILACC, IMR, IMR+, IMR2, ISA-HUB, MACE, Magic Packet, PCnet, PCnet-*FAST*, PCnet-*FAST*+, PCnet-Mobile, QFEX, QFEXr, QuASI, QuEST, QuIET, TAXIchip, TPEX, and TPEX Plus are trademarks of Advanced Micro Devices, Inc.

Microsoft is a registered trademark of Microsoft Corporation.

Product names used in this publication are for identification purposes only and may be trademarks of their respective companies.