Design of the DEC LANcontroller 400 Adapter

By Richard E. Stockdale and Judy B. Weiss

Abstract

The DEC LANcontroller 400, Digital's XMI-to-Ethernet adapter (DEMNA), connects systems based on the Digital XMI bus to an Ethernet/IEEE 802.3 local area network (LAN). These systems use the XMI bus either as the system bus (VAX 6000 systems) or as an I/O bus (VAX 9000 systems). The new systems, which can utilize the full bandwidth of the Ethernet, are characterized by increased host processor speeds. The DEMNA adapter was designed to support these I/O requirements. In addition, console and monitor facilities were built into the adapter firmware for debugging, verification, and user visibility. The adapter's performance for small packets exceeds system capabilities, and Ethernet bandwidth is the limiting factor for large packets.

The high-performance DEC LANcontroller 400, Digital's XMI-to-Ethernet adapter (DEMNA), connects systems based on the Digital XMI bus to an Ethernet/IEEE 802.3 local area network (LAN). These systems use the XMI bus either as the system bus (VAX 6000 systems) or as an I/O bus (VAX 9000 systems). It is an intelligent adapter that implements the physical layer and part of the data link layer of network protocol. The term intelligent refers to the packet processing performed by the adapter as part of the data link layer.

The DEMNA adapter was needed to support the I/O requirements of the VAX 6000 and VAX 9000 systems, which can utilize the full bandwidth of the Ethernet. The adapter also provides the ability to configure these systems without a BI bus. For these systems, the DEMNA adapter is the only Ethernet connection available.

The DEMNA adapter is controlled by a port driver that resides in host memory. The interface between the port driver and the DEMNA firmware (the port) is a ring-based design which is optimized

adapter (DEMNA), connects a system based on the Digital XMI bus to an Ethernet/IEEE 802.3 local area network (LAN). This adapter is intended for Digital systems that use for low system overhead and high performance.

The DEMNA adapter has the following major features:

- Supports Ethernet/IEEE 802.3 protocols

- Supports up to 64 users (each one a separate protocol such as local area transport [LAT] software, DECnet network software, or clusters)
- Supports two modes of addressing: VAX virtual addressing and 40-bit physical addressing
- Allows buffer chaining on transmit
- Performs packet filtering and validation on receive
- Supports Digital's maintenance operations protocol (MOP) functions
- Provides support for diagnostic routines and field service functions implemented through the system console or diagnostic software
- Has console and monitor facilities that allow a console user to monitor DEMNA operation and network utilization

This paper begins with a logic overview of the DEMNA device. The sections that follow discuss the factors that influenced design and implementation, describe the major performance metrics and user visibility operations, and review the design results and future needs.

Logic Overview

The DEMNA adapter is a single-board XMI adapter based on complementary metal-oxide semiconductor /transistor transistor logic (CMOS/TTL) technology. As shown in Figure 1, the hardware consists of four separate subsystems:

- Microprocessor
- Direct memory access (DMA) and shared memory
- XMI interface
- Ethernet

The microprocessor subsystem contains the CMOS VAX (CVAX) processor, system support chip (SSC), boot read-only memory (ROM), Ethernet address programmable read-only memory (PROM), electrically erasable programmable read-only memory (EEPROM), but is copied to RAM for execution. The boot ROM contains the initialization code and diagnostics. This subsystem also provides a console interface through the SSC for diagnostics, module debugging, and network monitoring.

The DMA and shared memory

and random-access memory (RAM). The microprocessor subsystem provides an internal, high-speed CDAL bus so that the CVAX processor can fetch its instructions and execute them without being delayed by the other controllers on the module. The firmware is stored in EEPROM, subsystem provides the means of communication between the CVAX processor and the other subsystems. The devices arbitrating for this shared memory are the CVAX processor, the gate array, and the Local Area Network Controller for Ethernet (LANCE) chip.

The XMI interface subsystem contains the XMI network adapter (XNA) gate array and the XMI corner. The XNA gate array is the data-move engine for the DEMNA adapter and contains all the XMI-required registers.

The Ethernet subsystem contains the LANCE chip, the serial interface adapter (SIA) chip, and various bus interface logic modules. The Ethernet subsystem receives packets from the Ethernet and stores them in the shared memory. When transmitting a packet on the Ethernet, the LANCE chip gets the packets from shared memory and transmits them on the Ethernet.

Design

The design of the DEMNA adapter was influenced by many factors, including previous adapter design experiences, available hardware such as Ethernet chips, and system requirements. The DEMNA team was assigned the following tasks:

- Produce a working Ethernet adapter that could be used by operating systems such as VMS, ULTRIX, ELN,
- Deliver high performance, measured by the amount of Ethernet bandwidth supported at various packet sizes, with minimized host overhead
- Supply debugging features for design verification and field maintenance of the adapter

First, we reviewed previous adapters to determine what improvements could be made. We learned that a complex host interface complicated host software and adapter firmware and greatly affected performance. One of these adapters, the Digital BI Ethernet Network Adapter (DEBNA), implemented a generic port interface that used interlocked queues containing a queue entry with a buffer name that indexed into a buffer descriptor table (i.e., an additional level of indirection). In addition to the firmware complexity, the hardware was not well suited to a complex port interface.

Another area in which improvements could be made over previous Ethernet adapters was the amount of processing performed by

and custom operating systems on hardware configurations that use the XMI bus as a system bus or an I/O bus

the host processor during receive packet filtering, address translation, and buffer copies. Overall system performance improves

if this processing can be reduced by performing part or all of these functions in the adapter. This difference transforms the

adapter from a dumb adapter (much of the data link processing performed by the host) to an intelligent adapter (much of the processing performed by the adapter).

The results of our analysis of older Ethernet adapters led us to choose a design that employs a simple host interface, off-loads the host whenever possible, uses rings instead of queues, and supplies the address of the buffer directly with the ring entry rather than indirectly through another data structure.

The design of the adapter was now consistent with the needs of the new VAX 6000 and VAX 9000 systems. These systems, characterized by increased host processor speeds, needed increased I/O performance. The task of the DEMNA team was to fill that need for Ethernet I/O.

Type of Adapter

The DEMNA product is a store-and-forward adapter, i.e., it copies data to and from host memory by way of temporary storage on the adapter. This data transmission differs from that of a cut-through adapter in which data

We designed a simple host interface, using rings instead of queues. Interrupts to the host were kept to a minimum, from one interrupt per packet at light loads to a fraction of that number under heavy loads. As seen in Figure 2, the port and the port driver (host) share the following data structures, which reside in host memory:

- Port data block. This structure gives the port the location of the rings and page tables in host memory and is a repository for error information.
- Command and receive rings. These rings contain information describing outstanding command and transmit requests and buffer information for receive buffers.
- Transmit, receive, and command buffers. These buffers contain packet data and command data.

These data structures constitute the primary means of communication and data transfer between the port and the port driver. Control status registers (CSRs) are provided for port poll demand registers, XMI context, and port

flows directly between host memory and the transmission medium. However, the DEMNA adapter is actually able to gain some of the benefits of cut-through on the receive side.

Host Interface

initialization.

Two rings are used in the host interface: the command ring and the receive ring. Each ring consists of 1024 bytes of physically contiguous memory, and each ring contains entries that

describe a buffer or a set of buffer segments (when chaining transmit buffers). The number of entries in the receive ring is fixed, since each entry points to a single contiguous buffer. The size of each transmit ring entry is variable and is fixed at initialization time.

The port and port driver process the entries in each ring in sequential order, starting with the first entry. A ring entry can be processed only by its owner. When the last entry in the ring is reached, processing starts again with the first entry.

Host interrupts are minimized by using a ring release function, which counts the number of ring entries processed for completion by the port and the port driver. The port driver counts the number of completed entries and writes this count to a completion CSR when it has finished processing all the completed transmit and receive ring entries. The port maintains the same count and issues another interrupt whenever it sees that its count and the count last written by the port driver are different. This function ensures that the port driver is interrupted only when it stops processing the rings because there is nothing else to process. The port driver can process multiple completed transmits and receives after each interrupt as well. Thus, no spurious interrupts are issued and the number of interrupts is reduced by processing multiple completions at once.

Adapter Design

The firmware is written in VAX MACRO code. An alternative was to use MACRO for the transmit and receive paths and a higher-level language for initialization, shutdown, and error handling.

CVAX RAM (used by the CVAX processor exclusively) consists of 256 kilobytes and contains the firmware and data structures (the firmware is copied to RAM during self test). Smaller RAMs would have been slightly less expensive but would have complicated the firmware update procedure and limited the ability of the firmware to use the large data structures needed for receive packet filtering.

Shared RAM (shared by the CVAX processor and the LANCE chip) consists of another 256 kilobytes. This RAM contains the transmit and receive buffers as well as the LANCE transmit and receive rings. There is a vast amount of buffering space here, so the DEMNA device can tolerate a considerable amount of inattention from the host before being forced to discard incoming receive packets.

Erasable programmable read-only memory (EPROM) consists of 128K bytes for diagnostics and firmware boot code, including a backup copy of sufficient operational firmware to allow an update of EEPROM for initial load or subsequent update. EEPROM consists of 64K bytes for operational firmware,

However, this approach was not chosen because it complicates the interface and would have resulted in firmware size difficulties.

diagnostic patches, and error history data.

The gate array (data mover) handles the data move and quadword read/write operations. The data-move operations transfer buffers

between the host and shared RAM. The quadword read /write operations are used for control functions, such as reading ring entries, reading address translation information, and writing ring status on completion. Once the firmware initiates a data-move operation, other work is performed by the firmware while the data move progresses.

Interrupts are very costly; therefore, we chose to limit the number of interrupts fielded by the CVAX processor. A LANCE interrupt costs CVAX interrupt overhead, plus a LANCE CSR access, plus some normal interrupt overhead to save and restore registers. A data-move interrupt is less costly, but the firmware can be coded so that the data-move operation is usually complete, thus eliminating the need for the interrupt. Polling is performed for all LANCE- and data-move-related functions, but interrupts are used for local console I/O and error events.

Driver Design

The DEMNA team needed to design a driver that would be compatible with existing drivers but that would use all the features provided by the adapter. For VMS requests directly to the adapter.

For ULTRIX systems, the driver runs at a lower level with respect to packet filtering so it cannot take advantage of this feature. However, buffer chaining is used on the transmit side. As a transmit request traverses the various software layers, it accumulates buffer segments which the driver has to concatenate into a transmit frame. To avoid buffer copies in all but the extreme and infrequent cases, the driver then passes up to 11 buffer segments to the adapter.

To allow customer-written drivers for special applications, we documented the interface to make it readily available to customers.

Debug Tools

The adapter has a very simple mission in life: to transmit and receive packets. To verify operation, some debug tools are needed. The goal for the DEMNA team was to provide extensive debug tools both in the operational firmware and in standalone user tools. This design would allow debugging and verification

systems, this meant using the set of common routines that provide much of the data link functionality of the driver, but avoiding packet filtering. Another goal was to limit the copying of data by passing in the development lab and in other, less-controlled environments. These debug tools are discussed further in the Visibility section.

Implementation

This section describes the implementation of the DEMNA adapter through its major functional blocks:

- Scheduler
- Port processing
- Command processing
- Transmit task
- Receive task
- Console task
- Monitor task

Scheduler

The scheduler is a round-robin routine that simply checks for work, does it, checks for work, does it, etc. There are no context switches, but some context is maintained in registers and shared by all routines. The scheduler, when idle, consists of about 18 VAX MACRO instructions. Transmit and receive tasks are given higher priority by duplicating their scheduler entry. When not idle, one pass of the scheduler processes four packets.

Port Processing

Port processing controls adapter initialization and shutdown, LANCE initialization and restart, fatal adapter error handling, gate array error

The command ring usually contains transmit buffers, which can contain commands for special functions. These commands are included in the command ring to allow the port driver to synchronize control requests with transmit requests, e.g., user startup and stopping.

Command processing routines are called by the transmit task after the command buffer has been read from host memory. The commands consist of user startup (consisting of user context such as protocol type, packet format, physical address to use, and multicast addresses to enable), user stopping, read counters, and a set of maintenance commands.

Transmit Task

The transmit task copies a packet from the host memory to adapter buffer memory and tells the LANCE to transmit it onto the Ethernet (store and forward). After the LANCE has completed the request, the firmware writes transmit status to the command ring entry, signifying completion of the transmit.

To minimize service time, the code in the

handling, and miscellaneous host interface functions. This task also handles firmware updates of EEPROM. Command Processing	transmit path was carefully scrutinized. The number of checks and branches was minimized for the optimized path. The optimized path through the transmit code is the 30-bit virtual addressing path, which is the most used. However, the 40-bit

physical addressing path still results in better throughput because this path does not require any address translations, which are timely. The instruction sizes were shortened when possible, using word instructions instead of longword instructions, to reduce the amount of instruction prefetch by the CVAX processor. Routines were placed on quadword boundaries to maximize cache efficiency. When waiting for data moves to complete (getting the transmit buffer from host memory) or obtaining address translation information from the host, the firmware was designed to perform other functions to increase the probability that the operation would be performed when the firmware needed it.

Receive Task

The receive task has the simple job of handing received packets to the port driver. This task is complicated by the need to off-load the host of part of receive processing (including packet filtering, packet validation, maintenance of counters, and processing MOP messages) and to make duplicates of packets when more than one user has requested a copy. It is in small groups (192 bytes) to allow the benefit of cut-through on larger packets.

Packet filtering is done for the destination address and for user type, either protocol type for Ethernet, destination service access point (DSAP) field for 802, and protocol identifier (PID) value for 802 subnetwork access protocol (SNAP) packets. Additional filtering is done for users who request all traffic or all multicast traffic. Filtering is done by maintaining a 64-bit user mask, which accumulates the list of users who want a copy of the packet according to the characteristics of the packet and what each user has requested.

Packet validation consists of length checks for Ethernet frames (if the user is using a length field after the protocol type) and for 802 frames. This saves the driver a little work. Additionally, users can request only packets smaller than a selected size; the adapter discards packets that exceed this size.

The cut-through feature adds complexity and reduces throughput on small packets, but provides many benefits for larger

further complicated by the need to provide buffering, which the port driver uses to prevent the driver from supplying large numbers of buffers. For enhanced performance, the firmware deals with receive packets packets. When a packet larger than 192 bytes is received, the packet filtering and validation of all but the length is done for the first segment. This segment is then copied

into the host buffer, and subsequent segments are copied appropriately. The last segment completes the packet validation and cyclic redundancy check (CRC). The difficulty occurs when the packet validation fails or an error is detected, because the packet is discarded and the context for the now-free receive buffer has to be restored. The firmware elects to save as little context as possible for each packet and to regenerate buffer context after the error, i.e., fetching the ring descriptor anew and redoing the address translation.

Console Task

The console task accepts and parses console commands and displays the requested data. There are two means of accessing the console: local and remote. The local console is accessed by a terminal connected directly to the DEMNA adapter. The remote console is accessed through MOP console carrier commands directed at the adapter from another system. A remote console may also be used to access a DEMNA device on the local system (coming in through transmit instead of receive). The firmware does not distinguish between transmit or receive

where it is formatted and displayed on the screen.

Due to code size limitations in the EEPROM, compressed versions of the console screens are stored in the EEPROM. At initialization time the screens are uncompressed and stored in the RAM. (The screen compression saved 5 kilobytes in the EEPROM.) To easily setup and maintain the screens, especially since they often changed during the project, the screens were set up in separate text files. The fields in the screen were coded with different data types, such as date or longword. The screen was then put through a PASCAL program to convert it to a VAX MACRO data structure and compress it.

The local console and the remote console can be run simultaneously. They have separate input and output buffers, the same decode and formatting code, and different input and output methods.

The remote console uses the MOP console carrier, coming in on transmit or receive. The command/poll and response/acknowledge commands are sent by the MOP program, i.e., either the network control program (NCP) or a user program that implements

operations from remote consoles. The console block accepts the commands and decodes them, and the monitor block determines the status. The monitor block passes this status back to the console block the MOP console carrier. The console code extracts the input characters from the command/poll packet and returns a response /acknowledge packet with any available data from

the remote console output buffer. When a command has been entirely received, it is decoded and executed and the response placed in the remote console output buffer, which is sent back to the user in response /acknowledge packets.

The local console is a terminal directly connected to the DEMNA device and interfaced through the SSC universal asynchronous receiver transmitter (UART). This terminal connection receives and transmits one character at a time. Characters are collected into the local console input buffer and complete commands are parsed and executed. Response data is placed in the local console output buffer. The local console uses interrupts to signal when a character has been typed or when the UART is ready to transmit another character. These are the only interrupts used on the module, except for error interrupts. Since console interrupts are relatively infrequent, they are less costly than polling.

Monitor Task

The monitor facility operates mainly during receive or transmit. It also runs as a low priority

Performance

As stated previously, the primary goal of the DEMNA adapter was high-speed performance, i.e., this adapter would not create a bottleneck when placed in a system. The major performance metrics we identified were throughput, service time, latency, and reliability.

- Throughput is the number of packets or bytes of packet data that can be transmitted or received per unit of time.
- Service time is the time a packet spends in each stage along its path from source through host software and driver, through adapter, over wire, through adapter, and through driver and host software to the destination.
- Latency is another measure of service time. It is a measure of delays encountered by queue depths of more than one at various points.
- Reliability is measured as the probability of packet loss under a receive load. It is also measured as adapter buffering and host buffer allocation

entry in the scheduler to deal with debugging and verification activities (when debugging firmware is enabled).

effectiveness. For some protocols, recovery from packet loss takes a significant amount of time, and the loss of a packet may be quite noticeable to a user. Hence, recovery is related to a user's

perception of reliable operation.

The performance goal of the DEMNA team was to minimize the service time through the adapter to maximize throughput. This is most critical for small packet sizes. If the service time is greater than the time it takes to transmit or receive a packet, then queue depths increase, increasing latency for subsequent packets. Small packets are critical because, obviously, they take less time to transmit or receive.

The speed of the Ethernet wire and the XMI bus must also be considered. The Ethernet operates at 10 megabits per second. The available bandwidth into memory and the capacity of the XMI are much greater; thus, the Ethernet is the limiting factor. To maintain maximum throughput, the DEMNA device must write and read packets to and from host memory at a speed equal to or greater than the Ethernet wire. If this speed is obtained, then the service time of the DEMNA adapter must be less than the time it takes to transmit or receive one 64-byte (small) packet to or from the Ethernet wire to maintain maximum throughput

The primary hardware factors influencing adapter performance are CVAX performance, DMA engine throughput, and bus contention.

The gate array DMA engine can sustain between 11.5 and 13.5 megabytes per second on a VAX 6000 system. When transferring packet data (and attendant host ring processing), the firmware can sustain about 5.8 megabytes per second. This is the approximate rate at which the firmware would deliver a burst of large packets that had been stalled due to a lack of receive buffers.

The CVAX chip used is the 60-microsecond variant (the same one used in the VAX 6000 Model 310 processor). As seen in Figure 1, the processor runs on its own internal CDAL bus which has RAM containing firmware and private data structures. Thus the processor does not contend for the same bus as the gate array and the LANCE chip. However, the CVAX processor does touch shared memory and gate array registers; therefore the possibility of contention is significant. Logic analyzer measurements indicate that about 14 percent of CVAX cycles are consumed while waiting

at all packet sizes. Hardware	for access to the shared memory bus for minimum size packets. For large packets the consumption is 33 percent, but the cycles needed are considerably less than the remainder. The effect on the gate

array accounts for part of the difference between the speeds of 11.5 to 13.5 megabytes per second and of the 5.8 megabytes per second mentioned above.

Firmware

Throughput is limited by the Ethernet bandwidth for packet sizes greater than 88 bytes. The average packet size on Ethernet is approximately 150 to 450 bytes per packet for a mix of DECnet, LAT, and cluster traffic. Table 1 represents the throughput that the host software can see, given sufficient host computes. These numbers show what might be expected. Virtual addressing costs some performance, and receive filtering accounts for most of the difference between transmit and receive.

Table 1

DEMNA Throughput

Packet Length (bytes)	Ethernet Maximum	LANCE Maximum	Transmit (microseconds)	Virtual Physical	Transmit Virtual	Receive Physical	Receive Physical
64	14880	14662	13181	14633	12468	12918	
72	13586	13404	12592	13361	12254	12830	
80	12500	12345	12247	12340	11813	12227	
88	11574	11441	11432	11438	11441	11441	
96	10775	10660	10656	10658	10660	10660	
112	9469	9380	9380	9380	9380	9380	
128	8445	8374	8374	8374	8374		

8374

256	4528	4508	4508	4508	4508	4508
512	2349	2344	2342	2344	2344	2344
1024	1197	1195	1195	1195	1195	1195
1518	812	812	812	812	812	812

It is interesting to look at the number of instructions executed by the CVAX processor for each receive and transmit packet as the measure of how much work must be done for each packet. These instruction counts are for minimum size packets in virtual address mode and increase slightly with increasing packet sizes.

For a transmit, the number of instructions required was about 134, consisting of 5 instructions for work

done in the scheduler to determine initial transmit context, 77 instructions for the data transfer from host memory, 18 instructions to get the LANCE chip to begin transmitting, and 34 instructions to process packet completion and to update status in the transmit ring entry in host memory.

For a receive, the number of instructions required was about 160, consisting of 5 instructions for work done in the scheduler to determine initial receive context, 40 instructions to deal with the LANCE operations, 20 instructions for packet filtering, 65 instructions for the data transfer to host memory (including some time spent finding a user and validating the packet length), and 30 instructions for the prefetch of the next receive ring entry.

Some throughput was traded off in the interest of reducing adapter-added latency. By processing receive packets in groups of 192 bytes, the latency contribution for any packet size is much smaller than it would be if all the packet processing occurs after the packet has been fully received. Thus the time between the end of a packet on the wire and the host interrupt is fairly constant from 64- to 1518-byte packets, 50 to 70 microseconds.

Reliability

Reliability, or probability of loss, is measured by how large a burst of traffic the adapter can withstand at the maximum receive rate and deliver these packets to the host without losing any. Adapter reliability was measured at various packet sizes. A burst of 5 seconds without packet loss was considered to be of "infinite" duration.

Table 2 shows that the DEMNA adapter can survive a significant burst of activity without packet loss. Such activity is unlikely, but possible, depending on the application being run and on the network configuration.

Table 2

DEMNA Receive Burst Tolerance

Packet Length (bytes)	Burst Virtual (packets)	Burst Virtual (microseconds)	Burst Physical (packets)	Burst Physical (microseconds)
64	3250	221661	3843	262106
72	5116	381677	11591	864741

Table 2 (Cont.)

DEMNA Receive Burst Tolerance

Packet Length (bytes)	Burst Virtual (packets)	Burst Virtual (microseconds)	Burst Physical (packets)	Burst Physical (microseconds)
80	9917	803321	Infinite	Infinite
88	Infinite	Infinite	Infinite	Infinite

This testing does not measure how host software performs buffer allocation for a user application or for the adapter as a whole. For the latter, the DEMNA adapter accounts for any lack of buffering by the host by not discarding a packet if a buffer is not immediately available. Instead, it waits up to three seconds for the host to supply a buffer.

Visibility

A system user looking at the operation of the network sees three areas of complexity: the system software, the network controller, and the network. When everything is working well, there is little need to look at any of these areas except and monitor facilities were built into adapter firmware from the outset; we knew that the visibility was crucial to adapter debugging and verification and would later be helpful to users.

System Operation

The console displays XMI utilization as apportioned among the XMI devices. This data comes from sampling done by the firmware of the "last XMI node active on the bus." From this, the user can estimate total XMI utilization.

The console also displays buffer occupancy on the adapter for transmit and receive, user configuration as to protocol type and characteristics, buffer availability counters, and host interrupt counters.

perhaps to predict future operation (by extrapolating network utilization or system usage) or to confirm that the system is indeed running well. When the system is not running well, visibility into these areas is crucial to understanding what is wrong and how to correct it. The console

This data indicates how the system is running, i.e., whether sufficient buffers are allocated to the device and to each user of the device. These counters also indicate how much attention the driver is paying to the adapter. For example, if the system is not tuned properly, the

adapter may be generating less than normal interrupts (because queuing delays are affecting the system operation). These queuing delays can be seen in the firmware counters, which monitor the depth of adapter queues and the ability of the adapter to give receives to the host, i.e., buffering on the adapter has been used to compensate for queuing delays in the host.

Adapter Operation

When the adapter is not malfunctioning, visibility into adapter utilization is important. The console displays program counter (PC) sampling results for the firmware, showing how busy the adapter is and where time is being spent. When looking at the I/O subsystem as a whole, it is important to know how much the adapter is contributing to queuing delays, buffer occupancy, and added latency. This adapter operation can be seen by looking at how busy the adapter is and how many buffers it has outstanding.

For adapter failure or problems on the XMI, the console displays error information which has been saved in EEPROM. This error data consists of fatal error context, data transfer or XMI error

The DEMNA device normally sees all packets on the wire (excluding packets less than 64 bytes in length [runt packets] and collision fragments). When looking at the adapter operation through the console facility, the user sees current network utilization and network error information. For transmit errors, the console displays the number of errors and date and time of the last occurrence. For receive errors, the console displays the number of errors, date and time, source address, and protocol type. Additionally, receive errors that are not counted (because they do not pass receive filtering) are displayed. For example, error information is displayed for a node generating packets with CRC errors regardless of the destination of these packets.

The console also provides the command SHOW NETWORK to display network utilization in node addresses and protocol types. For this command, the receive firmware calls a monitor facility routine for each packet seen on the wire. This routine maintains statistics for each source and destination node address, consisting of

context, and results of self-test.	the number of packets and the number of bytes. At
Network Operation	three-second intervals, the console calls a monitor routine which adds statistics over the prior interval to cumulative data

for each node, collects top nodes and protocol data, and clears the interval data to prepare for the next three seconds of monitoring. Figure 3 represents a sample network monitoring display.

Debug Tools

The monitor task provided other debugging functions during adapter debugging and internal field test. These functions are not visible features in the finished product. However, they are extensions to the functionality and illustrate the benefits of visibility into the adapter. A user program, XNAMON, was written to access the following functions.

- Traffic generation. It is difficult to generate heavy loads on an adapter, particularly because of logistics. Other systems are needed with enough processing power to generate the load. Using the XNAMON program, only one system was needed. XNAMON was run on it to direct other adapters to generate traffic to another node with a particular packet size

  at a specified rate. Since traffic generation could be done regardless

- Packet tracing. This function allowed a node to scan the receive stream for packets with selected source and destination addresses and protocol types. Either the packet header or the entire packet was saved for matching packets. This function was used extensively during initial debugging for validating transmit functionality. Later it was used for validation of MOP and related functionality by creating trace files on a known good node. We then ran functional scripts through a test generator, which used the traffic generator on one node to send a test packet to the node under test. The command and the response were traced by the trace node and the test program collected the trace data and compared it against known good data. Packet tracing was also used to verify packet filtering by devising a test program that could start up particular user configurations and loop back any packets received.

- Adapter test. The ability to exercise a module under stress

of system state (except for power on), there was always a good supply of traffic generators.

was critical to adapter hardware verification. The functionality in question was the Ethernet subsystem and the XMI interface through the gate array. The monitor facility

provided this test functionality by doing MOP loopback operations to another node while doing various data transfer operations to host memory. Data compares were done on completed transactions to validate data integrity. The XNAMON program provided the interface for this function and the remote display of its results.

- Remote debugger. The access to DEMNA internals allowed remote adapter memory dumps and remote inspection of data structures while the adapter was running.

Conclusion

The DEMNA adapter meets the requirements of the VAX 6000 and VAX 9000 systems. In fact, the performance for small packets exceeds the capability of these systems. For larger packets, Ethernet bandwidth is the limiting factor. Our experience illustrates some advantages and disadvantages of choosing a firmware-based design over an interface implemented entirely in hardware.

Advantages of a Firmware-based Design

- The firmware can be changed easily (bug fixes or changes in functionality), thus reducing long-term maintenance and support costs. Also, changes can be made in the field by a firmware upgrade rather than requiring module rework at a manufacturing site.
- By designing in the firmware, designers can avoid software driver complexity and the necessity of hardware redesign.
- The firmware can provide powerful debugging mechanisms and tools.
- The firmware is very flexible. Changes to support hardware problems or additional off-load of host computes can be considered late in the design cycle. This may also allow new port architecture and addressing changes for creating new products.
- Firmware designs allow extensive functionality for lower product and development cost than a total hardware design.
- Firmware designs allow the hardware to be released earlier in the

The advantages of designing an adapter in firmware are as follows:

- The firmware can usually off-load host computes by doing more pre-processing.

development cycle.

Disadvantages of a Firmware-based Design

The disadvantages of designing an adapter in firmware are:

- The adapter is generally more expensive, considering the cost of a microprocessor subsystem with enough computes for the job.
- The adapter is slower in terms of latency. Some applications may be more sensitive than others, given the same throughput, but may have slightly larger service times per packet. The effect can be viewed in terms of buffer occupancy: an adapter with lower latency may utilize, on average, few buffers.
- The approach is not feasible for transmission media much faster than Ethernet, because the performance requirements of the microprocessor become extreme or the hardware assists for the microprocessor become too complex and costly.

Future Directions

Several characteristics distinguish future anticipated system design from current systems (such as the VAX 6000 and VAX 9000 systems).

Increased host processor speed moves the I/O bottleneck from the host to the I/O subsystem. To supply the I/O needs, the I/O subsystem must provide faster media, e.g., fiber distributed data interface (FDDI) in the near term, or multiple connections to slower media (such as Ethernet). The I/O adapters will be expected to provide significantly greater throughput with a smaller adapter contribution to latency. The effective performance of the system will be more sensitive to latency. For example, an application using a single threaded command/response protocol is extremely dependent on the amount of service time through the I/O subsystem at each end. As the processing speed increases, application overhead is reduced and throughput becomes dominated by the service time of the adapter and the transmission time.

Faster processors place a greater burden on the system bus and I/O interface, which necessitates a simpler bus protocol. This might consist of eliminating costly functionality such as byte masking and

	interlocks. However, a simpler interface to the I/O adapter will require considerable change to the port protocol to ensure its efficiency.
o Increased host processor power	
o Simplified bus design	
o Increased I/O bandwidth requirements	

The characteristics needed in future adapters are as follows:

- Greater throughput. This means more connections to a slower medium, such as a single adapter supporting multiple Ethernet connections.

  Or it means a faster medium. Additionally, configurations using Ethernets as point-to-point links will be more common, thus implying a heavier load on each Ethernet.
- Simpler host interface. This is necessitated by the simpler bus protocol. Bus overhead should be minimized, which includes the

  elimination of such functionality as page table access for virtual address translation. Also, the bus transfer size used by the adapter should be compatible with the basic data size of the system to avoid cache thrashing and unnecessary read-modify-write transactions.

- Reduced latency. The adapter should minimize its contribution to transmit and receive latency. This may mean reducing some of the functions done by an intelligent adapter

  length validation, and maintaining counters data. Improving packet filtering by host software would eliminate the reason for placing this function on the adapter in the first place.

  Filtering in host software is considerably more difficult than in the adapter. The difficulty comes from the need to deal with extreme user configurations. The DEMNA is bounded by limiting the number of users and node addresses. The extreme cases must still be done by host software.

Acknowledgements

The authors would like to acknowledge the following members of the DEMNA design team: Barbara Aichinger, Keith Bilafer, Mark Cacciapouti, Don Dossa, Linda Duffell, Bernie Hall, Jeff Huber, Helen McGreal, Jonathan Mooty, Dave O'Keefe, David Oliver,

Brian Parr, Art Singer, Andy Stewart, Fred Templin, Vicky Triolo, Ed Tulloch, and Don Villani.

General References

on receive, in order to speed delivery to the host after packet reception is complete. These functions include packet filtering, handling of maintenance operations packets,

Programmer's Guide (Maynard: Digital Equipment Corporation, Order No. EK-DEMNA-PG-001, 1990).

DEC LANcontroller 400 Console User's Guide (Maynard: Digital Equipment Corporation, Order No. EK-DEMNA-UG-001, 1990).

D. Mirchandani and P. Biswas, "Ethernet Performance of Remote DECwindows Applications," Digital Technical Journal, vol. 2, no. 3 (Summer 1990): 84-94.

D. Boggs et al., "Measured Capacity of an Ethernet: Myths and Reality," Proceedings of SIGCOMM '88 (ACM SIGCOMM, 1988): 222-234.

The Ethernet: A Local Area Network, Data Link Layer and Physical Layer Specifications, Version 2.0 (Digital Equipment Corporation, Intel Corporation, and Xerox Corporation, Order No. AA-K759B-TK, 1982).