An Overview of the Common Node Software

By Paul W. Ciarfella, David Benson, and David S. Sawyer

Abstract

To address an aggressive fiber distributed data interface (FDDI) program schedule and reduce the complexity of the concurrent development of multiple FDDI products, Digital developed a common implementation of the FDDI station management standard. This implementation, called the Common Node Software, manages the physical and logical connections to the 100-megabit-per-second fiber-optic ring for Digital's FDDI product set. Including the Common Node Software in each product yields consistent behavior at the FDDI data link and physical layers. Direct reuse of the software reduces the development and testing efforts by providing a proven implementation.

Introduction

The fiber distributed data interface (FDDI) data link provides clients with a connectionless to transfer information across the network.

The Common Node Software (CNS) implements the part of the FDDI station that controls and monitors connections within the FDDI network. To provide this service, CNS implements the protocols defined by the FDDI station management (SMT) standard plus Digital's value-added enhancements and, in addition, manages the services provided by the FDDI chip set. These services include the ring memory controller (RMC), media access control (MAC), and elasticity buffer and link management (ELM) chips.[1,2,3] CNS does not provide International Standards Organization (ISO) 8802-2 logical link control (LLC) support, which defines how data is reliably exchanged between two communicating systems. Although not provided by CNS, LLC support is included in each member of Digital's FDDI product set. Figure 1 illustrates

data transmission and reception service. An essential element of this service is reliable connection to the physical network, allowing clients

the functional requirements fulfilled by the Common Node Software.

An Overview of the Common Node Software

In this paper, we begin with a discussion of events leading to the development of CNS. Next, we present a detailed functional description of the FDDI station management and chip management services and specifics of the core and external libraries of the common code. We then describe the CNS development effort. A summary of the testing process follows, including details of the design verification test (DVT) monitor, developed by the team, and interoperability testing among FDDI vendors. Finally, the benefits of common code realized by the project team are discussed.

Background of the CNS Development Effort

When development efforts for the DECbridge 500 and DECconcentrator 500 firmware began, the American National Standards Institute (ANSI) FDDI MAC, physical layer (PHY), and physical medium dependent layer (PMD) standards were complete, but the station management specification was not. The SMT draft was frequently changing; new protocols were defined and modifications to existing protocols were made with each meeting of the X3T9.5 SMT working group, which is

These changes complicated the concurrent development schedules of the DECbridge 500 and DECconcentrator 500 data link software. If two independent firmware teams designed their own FDDI data link software, both teams would need to follow the development of the SMT draft. These efforts could result in different interpretations of the SMT protocols, of how Digital's FDDI chip set works, and of the functions data link software should and should not perform. Producing multiple, independent SMT implementations could lead to incompatible products that exhibit inconsistent behavior and are unable to communicate.

During the evolution of the SMT draft, the physical connection management (PCM) protocol pseudocode defined by the draft changed often; some changes caused previous versions to be incompatible. PCM is responsible for the management of full-duplex physical connections between two FDDI PHY entities. The PCM pseudocode defines a synchronized bit-signaling communication process between two

responsible for developing
the standard.

connecting nodes to exchange connection information. If two nodes attempting to connect to one another execute incompatible versions of the PCM pseudocode, these nodes will not connect. To avoid this scenario, and possibly others, a decision was made to produce a common, reusable implementation of the SMT protocols and FDDI chip management. The initial goal of the CNS project team was to constrain the domain of possible SMT-related problems which could appear during the development of the FDDI product set.

Another important goal of the CNS project was compliance with both the SMT standard and the Digital Network Architecture (DNA) FDDI data link functional specification. Compliance with the SMT standard could increase the probability that Digital's FDDI products would interoperate with those of other vendors. Compliance with the DNA architecture would guarantee interoperability among Digital's current and future products.

The CNS team worked closely with Digital's representatives in the ANSI began. The end result is a set of reusable software libraries which provide the functions necessary to implement the SMT protocols and to manage the FDDI chip set on each FDDI product. This reusable code, called the Common Node Software, is shared by the DECbridge 500 and DECconcentrator 500 products and also by more recently introduced FDDI products, such as the DECcontroller 700 adapter.

As the CNS design matured, the advantages and benefits of a common code implementation became more apparent to the FDDI program team. Originally, the design constrained CNS to operate under the same operating system used in the DECconcentrator 500 and DECbridge 500 products. CNS was extended to support microprocessor and operating system independence to accommodate future FDDI products, thus facilitating the portability of CNS to other environments.

This extended CNS support allows a variety of implementations, including host-based network controllers and firmware-resident drivers. Consequently, Digital is benefiting by distributing the software

X3T9.5 FDDI SMT working group to develop the functional requirements of the software. As the SMT specification evolved, the functional requirements were completed and the development of the software externally through third-party licenses to promote rapid introduction of quality FDDI products to the expanding marketplace.

An Overview of the Common Node Software

Functional Description

This section presents details of the station management standard services and the management of services provided by the FDDI chip set. A functional block diagram of the Common Node Software is shown in Figure 2.

Station Management

The station management standard defines the services used by every station in an FDDI ring to monitor and control both the station itself and the state of the ring. The functionality defined by the standard includes connection management (CMT), ring management (RMT), and SMT frame-based services.

Connection Management CMT is primarily responsible for the maintenance of physical connections to the FDDI ring. This involves initializing and establishing connections between physical layer port (PHY port) entities and configuring the internal data path of a station. CMT is divided into the following three areas: entity coordination management, which coordinates the trace process and manages the optional optical bypass function; configuration management, which manages the configuration of the PHY and MAC entities within a station; and PCM, which manages the physical connection between a station's PHY and the PHY of the adjacent station. Table 1 lists the functions

and divisions of CMT and summarizes the value added by Digital.

Table 1

Connection Management Functionality

Name	ANSI SMT Standard	Value Added	Description
Connection Management	Yes		Manages physical connections and station configuration. Contains entity coordination management, configuration management, physical connection management, and link error monitor.
		Yes	Enhances topology management.
Entity Coordination Management	Yes		Coordinates the trace process and manages the optional optical bypass function.
		Yes	Enhances trace function to be insensitive to network reconfigurations.
Configurat Management	Yes		Manages the configuration of the PHY and MAC entities within the station.
		Yes	Function is integrated within hardware.
Physical Connection Management	Yes		Manages the physical connection between connected PHY entities.
		Yes	PCM is implemented in hardware.
Link Error Monitor	Yes		Monitors active physical connection quality.
		Yes	Additional errors are recognized above

those_required_by_the_standard.

CMT provides a link confidence test and a link error monitor to check the quality of the physical link. When a connection is formed, the link confidence test invoked by PCM is performed to determine if the quality of the connection is adequate for proper ring operation. If the test fails, the

connection is not allowed to form.

The link error monitor checks the quality of the connection after it forms by computing and monitoring the link error rate to determine if this rate is acceptable. If the rate falls outside the acceptable range, the connection will be terminated. A value-added feature of the link error monitor is error detection exceeding that required by the SMT standard.[3]

CMT is also responsible for topology control on new connections. Topology control, implemented by CNS as part of the PCM pseudocode processing, enforces a set of connection rules defined to prevent the formation of illegal ring topologies. The topology rules utilized by CNS are more strict than the default rules suggested by the SMT standard. Connections that are clearly misconfigurations are not allowed to form.

CMT also provides support for the trace function, which is a recovery mechanism for the loss of logical continuity of the ring. The trace function is initiated by the station downstream from the logical break causing the break should fail its test and not rejoin the ring. Another value-added function in CNS is the enhancement of the trace algorithms to ensure proper termination of a trace even in the presence of simultaneous network reconfiguration.

Connection management is implemented in both hardware and software. CNS provides the CMT services that must be implemented in software but are not performed by the ELM chip. For example, the PCM state machine, implemented in the ELM chip, specifies the timing, state, and physical bit-signaling used in the connection process. The PCM pseudocode, on the other hand, is implemented in CNS. This pseudocode is used to communicate connection information between neighboring PHY ports. Coordinating the PCM state machine and pseudocode provides full PCM functionality as defined in the SMT standard.

Another example of CMT services provided by CNS is the link error monitor. The ELM chip provides facilities to detect and signal the occurrence of bit errors on the link. CNS computes the link error rate based upon the data

and is propagated upstream toward the break. Stations that receive the trace notification leave the ring and perform a diagnostic fault test in an attempt to locate a faulty MAC or data path. The station from the ELM and determines whether to sustain or to break marginal connections.

Ring Management RMT, also implemented in CNS, monitors the state of the logical link by using logical link status information received from the MAC sublayer. The information is then passed on to network management, such as Digital's extended LAN management software (DECelms), or to the LLC sublayer, for example, to report that the link is available for transmission service.

This MAC information is also used by RMT to detect and to resolve faults such as duplicate addresses and stuck beacon conditions. A stuck beacon condition occurs on the ring when a station's MAC continuously transmits MAC beacon frames due to a fault condition on its data path. The logical ring does not form because the MAC is not able to receive its own beacons. Detecting the stuck beacon condition, RMT sends special MAC frames, called directed beacons, onto the ring to notify other stations of the condition. After sending these directed beacons for approximately 370 milliseconds, RMT tells CMT to initiate the trace function. When the trace function successfully completes, the data path fault is detected and the

these stations with duplicate addresses strip each other's frames from the ring, causing ring instability and increased packet loss. If the duplicate stations are performing ring initialization, however, successful ring initialization may be prevented and the ring will not become operational.

The RMT state machine can detect only the duplicate condition of its own

MAC address. When this condition is detected, RMT has the option of removing the MAC from the ring, changing its address to a unique address, or forcing its MAC to lose the claim process and allow the ring to become operational. With the first option, a special control frame, called a jam beacon, is sent directly to the duplicate address to inform all stations with the duplicate address of the condition. LLC service associated with the MAC is disabled while the duplicate condition is present.

RMT duplicate address detection is complemented by the neighbor notification protocol. This protocol allows a station to learn both its downstream and upstream neighbors' addresses. By transmitting a periodic

logical ring is formed.

If two or more stations have identical MAC addresses, the MAC protocols are adversely affected. If the ring becomes operational, neighborhood information request frame (NIF) to its downstream neighbor, the protocol is able to detect from the received NIF responses that its MAC address is duplicated

on the ring. LLC service associated with the MAC is disabled until the condition is removed.

Station Management Frame-based Services

Station management frame-based services include both mandatory and optional functionality. The mandatory functionality includes the NIF, status information frames, echo frames, request denied frames, extended service frames, status reporting frames, and the as yet undefined resource allocation frames. Parameter management frames are defined as optional functionality.

The set of SMT frame-based services supports higher-level network management functions which are not part of SMT. The information provided by the services helps network management to determine the topology and state of the ring and to control the network. The status report frame service announces status information to network management. The optional parameter management frame service facilitates remote management of FDDI stations. The status information frame service provides station configuration and operation parameters. The echo frame service provides station-to-station, loopback testing using SMT frames. Request denied frames are sent by a station in response to receiving service requests that are not understood or are not implemented. Extended service frames allow the use of vendor-defined frames. This service provides the capability for exercising new SMT frame-based services. These services are also useful for certain functions embedded within Digital's stations. These embedded functions utilize the SMT frame-based services and are summarized in Table 2.

Table 2

Embedded Functions Supported by SMT Frame-based Services

Name	ANSI SMT Standard	Value Added	Description

duplicate Address test	Yes	Periodic NIF requests inform neighbor of station's existence, get neighbor's address, and test for duplicate of this station.

Table 2 (Cont.)

Embedded Functions Supported by SMT Frame-based Services

		Yes	Sending another NIF request to our own address improves coverage over ANSI required test.
Ring Purger Elec-tionb	No	Yes	Digital's ring purger rids the ring of no-owner frames and fragments. Ring purger election is controlled by a distributed algorithm.
SMT Gateway	No	Yes	Management may use any station as an agent to query other stations for SMT information.

Table 3

Table 2 Embedded Functions Supported by SMT Frame-based Services

Name	ANSI SMT Standard	Value Added	Description
duplicate address test		Yes	Periodic NIF requests inform neighbor of station's existence, get neighbor's address, and test for duplicate of this station.
		Yes	Sending another NIF request to our own address improves coverage over ANSI required test.
Ring Purger Elec-tionb	No	Yes	Digital's ring purger rids the ring of no-owner frames and fragments. Ring purger election is controlled by a distributed algorithm.
SMT Gateway	No	Yes	Management may use any station as an agent to query other stations for SMT information.

CNS makes use of the extended frame service by implementing the ring purger election protocol, which supports the ring purger function. The ring purger is one of several functions defined by Digital's FDDI architecture that add value to the SMT standard.[2]

The ring purger election protocol is an algorithm implemented within CNS; the purging function is implemented in the MAC

participating in the algorithm in order to communicate with each other. The station that either wins the claim process or has the highest address becomes the ring purger. Once elected, a ring purger enables the purging in the MAC chip and sends out periodic "hello" messages to the ring. If these messages are not received after a period of time, the election process is repeated. Other error control and recovery procedures are built into

chip. This distributed algorithm elects one station on the FDDI ring to be the ring purger. Candidate ring purgers, using the SMT extended frame service, send ring purger election frames to a multicast address known only to stations the algorithm to increase robustness.

Another value-added feature of CNS is the SMT gateway protocol. This protocol helps build ring maps on network management consoles. A ring map is a database that can be

used to build a graphical representation of the network topology. The map not only presents a visual image of the network, but also displays characteristics about each node on the ring such as its type and number of ports. The SMT gateway within CNS uses status information and neighbor information request frames to solicit information about other stations on the network. Responses received include information about a station's configuration, its network address, current counter values, and FDDI timer values. The gateway collects these responses and returns them, using the management protocol within the product, to the host management station building the map.

By providing a well-defined and protocol-independent interface to its clients, the SMT gateway can be used with any network management protocol. The first product to utilize the SMT gateway is DECelms software.[4]

FDDI Chip Management

CNS manages the services provided by the FDDI chip set, including the initialization of all configurable chip parameters such as timer values and interrupt masks. Software control of these modes is supplied by a set of interface functions within CNS. Many of the SMT-related protocols, such as CMT and the ring purger algorithm, are implemented in both the FDDI chips and CNS. CNS controls the operation of these algorithms through its management of the chip set.

CNS also provides consistent FDDI chip fault management for all products using CNS. Compile time and system initialization options allow the product designers to specify, for each fault, whether it should be considered fatal or nonfatal to the system. Also, for each event, a threshold is set indicating the maximum number of times the event is to occur over a predefined period before any action is taken. When the threshold is reached and a fault is classified as fatal, CNS removes the station from the ring and notifies the firmware kernel. The kernel is then responsible for further action, such as rebooting the hardware or running a diagnostic test. If a fault is considered nonfatal, CNS notifies the kernel of the event, but the station does not exit from the ring.

In addition, the algorithm can disable a recurring

Special chip modes, such as various frame reception modes supplied by the MAC, multiple loopback modes in the ELM, and parity detection in the RMC, are also controlled by CNS.

event for up to one second. This provides flow control of fault events and allows other processing to continue. During lab testing of the DECconcentrator 500

prototype, broadcast packet storms caused receiver overrun conditions in the RMC memory controller. An interrupt event is associated with the overrun, so the event occurred continuously, using up all available processor time. This situation effectively disabled the product until the storm ended. The flow control method was devised to throttle events in such a situation. Once the occurrence rate of the event slows down, the throttling stops.

Implementation Specifics

When the CNS development began, some important questions arose. Many of the questions were about FDDI itself. Most of the questions dealt with issues concerning the structure of the reusable software, for example,

- What defines common code?
- What functionality would be common across all products?
- How do we handle common functionality that must be implemented on different hardware platforms with different interfaces?
- What requirements, such as consistent interfaces, need to be placed upon external hardware and software?

The CNS project team developed a coding standard to facilitate a high degree of portability. The standard provides, for example, a portable set of type definitions to ensure that a long

Common Code

CNS is code written entirely in the C language, which can be executed on different hardware platforms without modifying the source code. This common code includes a library of core functions, which are completely portable, and a set of external functions to provide services that cannot be implemented in the same way on all products. The CNS core and external library interfaces with the remaining system firmware are shown in Figure 3.

But implementations of unions and bit-field definitions among C compilers is inconsistent. The language allows the assignment of variable names to individual or strings of bits. Control

integer is always 32-bits wide and a short integer is always 16-bits wide. Other type definitions specify the size of control /status registers, which may be either 16 or 32 bits, depending upon the platform.

/status register bit manipulation is easy to perform using the combination of structure and bit-field definitions. But the ordering of bit assignments can change from compiler to compiler. While one compiler assigns

bit 0 to a bit-field declaration, another may assign bit 31. Correctly mapping bit definitions in software to their corresponding definitions in a hardware register cannot be guaranteed. Thus, the use of unions and bit-field definitions was eliminated.

The Core Library

The software is partitioned into two libraries containing core and external functions. The core library consists of a set of completely reusable functions. No source code changes are necessary to execute this software on different products and hardware platforms.

The core library provides the major functions of CNS, but relies on the external library to provide operating system and hardware support. The core provides a set of interface functions that manage the physical and logical connections to the FDDI data link.

The core implements many of the SMT protocols such as any CMT not performed by hardware, duplicate address detection, the SMT frame protocols, and Digital's ring purger election protocol. The core library also provides interrupt processing functions for these interrupt processing functions are facilities to manage chip faults, such as parity errors or overrun conditions signaled by the FDDI chips.

The External Library

The external library is based on a well-defined set of functions such as allocating a transmit buffer, starting a software timer, or enabling the FDDI data path scrub function used to clear frame fragments from the ring. This library provides the direct interface to the product's operating system, buffer management services, and hardware configuration.

Digital's implementation of the external library is further divided into two sublibraries. The first sublibrary consists of functions specific, but common, to Digital's FDDI product set and is completely portable throughout the set. The other sublibrary of functions is specific to each FDDI product and is not portable among them. These functions deal mainly with special hardware access and configuration management, such as inserting the MAC onto the physical data path internal to the DECconcentrator 500 product. The interfaces to these functions

the MAC and ELM chips. The external library provides the interrupt service routines (ISR), and these core functions are called from within the ISR. Built within

are the same for all products, but different hardware designs or the limited functionality in a product may require distinct implementations. For example, the

DECconcentrator 500 device has an internal data path that interconnects the MAC and PHY entities within the product. Logic surrounding the MAC chip can be used to insert and remove the MAC from the data path or to bypass the MAC. The external library for the DECconcentrator 500 firmware contains two functions to insert and bypass the MAC. In contrast, the DECbridge 500 product's MAC is always on the data path and does not need to bypass or insert the MAC; therefore, the firmware does not contain bypass or insertion functions.

To facilitate consistent memory access, low-level packet memory functions enforce network byte order in SMT frames. Network byte order defines the order in which bytes are transmitted and received. These functions perform 16- and 32-bit read and write operations when building and parsing frames. Other product-specific functions provide access to designated packet memory locations for generating special MAC-level control frames called directed beacons, which are used by ring management in special fault situations.

CNS. The Phy data structure contains information about a single PHY port. The Link data structure reflects the state of the logical link associated with the MAC, while the Station data structure maintains information about the general state of CNS.

All information about the state of CNS and SMT is contained within these data structures. Other firmware agents residing within the products, such as extended LAN management software responders, need only to look in a central location for management information pertaining to SMT. Global variables defined by CNS for its sole use are kept to a minimum.

The structures are linked to reflect the actual configuration and data path of the MAC and PHY entities within the station. This linking provides easy support for various configurations and for the execution of configuration-based protocols. For example, in the DECbridge 500 product, the CNS data structures consist of one Station, one Link, and one Phy connected to reflect the single attachment station (SAS) architecture. The DECconcentrator 500 firmware has one Station,

Data Structures

Supporting both libraries is a set of three data structures, Phy, Link, and Station, that store state, configuration, and counter attributes information pertaining to one Link, and 12 Phy data structures connected to form the configuration of the station. A six-port concentrator with a management board is represented by one Station, one Link, and six Phy data structures. An eight-port

concentrator without a management board does not contain a MAC, and, thus, is configured as one Station and eight Phy data structures.

The CNS Development Effort

The actual design and development cycle for CNS covered a six-month period. During this time the SMT draft standard went through many revisions; the concept of licensing the CNS code was introduced; and Digital's FDDI chip set underwent a second-pass design cycle.

The licensing effort required rethinking and making adjustments to the partitioning and structure of the design to accommodate several layers of interfaces and support functions in CNS. These requirements resulted in an additional four to six weeks of effort, but yielded the benefits of a code that could be used outside Digital and a more generic design to accommodate future product designs.

Of all Digital's FDDI chips undergoing second-pass design, the ELM chip required the greatest number of changes. The majority of these changes were due to the major redefinition of portion of the CNS code was performed on Digital's logic simulation system (DECSIM) test bed. This testing utilized one of the same test beds constructed by the ELM chip designers configured with two ELMs connected together.[2] This test bed utilized the behavioral chip models in order to speed execution. Additional routines were written to emulate resources normally provided by the operating system (e.g., timer services). Roughly one week was spent in simulation, two days of which focused on developing and debugging the environment. Testing included the initialization of a good connection as well as connections that resulted in topology rejects, link confidence test failure, and link error monitor failure. No bugs were found in the ELM design, however several coding bugs were discovered. Later, when the first DECconcentrator 500 hardware was available in the lab, and the operating system services were debugged and available, CNS PHY code required only one-half day to debug before becoming operational. Thus, prior simulation of the code was clearly beneficial.

The first product that utilized CNS was the

the physical connection management portion of the ANSI SMT draft.[3] To minimize risk to the chip development effort, simulation of the connection management

DECconcentrator 500 firmware. The integration, debug, and design verification process spanned an eight-week period. For subsequent

products, the duration of this phase was greatly reduced: two weeks for the DECbridge 500 firmware and one week for the DECcontroller 700 adapter. This reduction in time was, of course, due to the reuse of the core and external libraries. Only a small subset of CNS was unique for each product; hence, debug time was minimal. Also, as experience was gained in verification testing, and the related tools improved, the test process became more efficient.

Testing the Common Node Software

One fortuitous advantage to the structure of CNS is that the core functionality only needs to be tested exhaustively on a single platform. The partitioning of core and external functionality and the external requirements to which each product environment must adhere yield this advantage. Thus, the only testing necessary on an individual product is the initialization of CNS and the external interface between CNS and the system firmware. These product dependencies include SMT frame transmission and reception, status and error

nature of many of the algorithms made testing difficult. The complexity causes automation of the test process to be an extremely involved task. Some functions, such as RMT, can be tested only in a multiple-station configuration due to the distributed nature of the algorithms.

Another difficulty in performing the initial testing of CNS was the lack of visibility into the executing software. The DECelms product reports some information maintained by CNS, but most of the data used during testing is not visible to network management. Also, the DECelms product was being developed at the same time as CNS and was not ready for use. In addition, there was no global visibility into the ring. At the time, commercially available FDDI datascopes or analyzers that would have been used to view symbol streams on the fiber were not available.

To facilitate testing, the CNS team developed a tool referred to as the design verification test (DVT) monitor. This tool provides a detailed view into the operation of CNS as well as automated tests for

message passing, and some functions unique to the product.

The task of test and verification of the CNS core presented some interesting challenges. The complexity and distributed CNS interface functions. The tool also has the capability to insert faults into the ring and exercise many of the SMT protocols.

An Overview of the Common Node Software

The DVT monitor has two components: a connection to a universal asynchronous receiver/transmitter (UART), which provides serial communication between the product and a display terminal; and monitor software, which is layered on top of CNS. Thus, the monitoring and managing of CNS is achieved by out-of-band access via the UART. This access is necessary to perform testing of ring fault conditions. The monitor software is run at a lower priority than all other system components to leave the system timing or operation unaffected. This software provides both nonintrusive or "peek" and invasive or "poke" management capabilities. Password protection on a login screen prevents unauthorized users from disrupting the network. To use the tool, one must log on to an FDDI product running the test software.

As a nonintrusive tool, the DVT monitor provides passive monitoring of the network status and related events. The tool provides real-time monitoring of all physical (or port) and logical (or MAC) connections in the product. Status windows continuously display the state of all physical and logical such as the station's address. Configuration hardware within each product can also be changed to affect the operational state of the ring. Changing this hardware is especially helpful for introducing duplicate addressed stations, stuck beacon conditions, beacon/claim ring oscillations, and other anomalies into an operational ring in order to exercise RMT and the trace function. To analyze SMT frames, an SMT frame agent exists in the tool to generate and receive any SMT frame type, including frames not defined by the standard.

Prior to the development of the DVT monitor, the FDDI tester was utilized to generate and receive SMT frames to test the proper operation of the SMT frame-based protocols.[2] In later testing, the FDDI tester was indispensable in creating a variety of traffic loads on the ring to test the products' responses to traffic loads. The tester allowed an arbitrary mix of frames sent at a programmable rate over the FDDI ring and made it simple to characterize the responses of the products to a wide range of network traffic conditions.

connections to the ring.

As an invasive tool, the monitor can be used to insert faults and to exercise the ring. The tool can be used to easily change station parameters,

An Overview of the Common Node Software

FDDI Interoperability Testing

Interoperability testing among the FDDI vendors is important for many reasons. Comprehensive interoperability testing among the vendors both reinforces the correct interpretations of the standard and performs the more immediate goal of verifying the correctness of the implementations. The proper operation of all stations on the network both in normal operation and in response to fault conditions is necessary if the commercial marketplace is to fully accept FDDI.

The nature of a ring topology requires the active participation of each station up to the MAC sublayer. Each FDDI station must establish and maintain physical connections and repeat frames without error. If a single station does not repeat frames, ring connectivity is lost. The X3T9.5 standards committee anticipated faults that prevent normal ring operation and devised algorithms to deterministically resolve such conditions. The addition of fault recovery schemes devised for the FDDI system, while necessary to guarantee proper operation of the vendors' implementations- especially in the presence of fault conditions.

Every vendor must make certain that its implementation adheres to the functionality as defined by the SMT standard. Digital encouraged cooperative testing among the vendors and participated in testing with many vendors at Digital's FDDI development center in Littleton, Massachusetts, at customer sites, and at other vendors' locations.

Interoperability Test Methodology

A test plan was developed originally for design verification and, later, for interoperability testing between Digital's FDDI product set and products from other vendors. The test plan concentrates mainly on the interoperability of the FDDI data link, defined by the FDDI PHY, MAC, and proposed PMD and SMT standards.

The plan covers connection management, ring management, and SMT frame-based services and functions defined in the SMT draft standard. The intent of the plan is to verify plug-and-play

network, made the standard more complex. Correct implementation of these complex protocols and distributed algorithms is essential to ensure that one vendor's implementation will operate correctly on the same ring with other capability as well as to correct operation under both normal and aberrant network conditions.

Connection management testing covers the physical connection management and configuration management processes. PCM testing covers the bit-signaling and connection initialization algorithms, the link confidence test and link error monitor, and verification of the connection matrix defined in the SMT draft. CFM testing verifies the correct operation of the reconfiguration scrub and MAC insertion functions.

Ring management testing covers duplicate address detection, including stuck beacon detection and recovery, directed and jam beacon initiation and reception, and the trace function. Other miscellaneous testing monitors the abusive use of restricted tokens and extended service frames.

Frame-based testing covers all required SMT frame protocols. These protocols are tested extensively for compliance to the SMT draft. Parameters within the frames are examined for consistency and correctness. For example, all timer values presented in SMT frames are verified to be in two's complement form, and all canonical addresses are correctly converted to FDDI most

The interoperability testing uncovered problems in many vendors' implementations, including Digital's.[5] Many of these problems resulted from inconsistent interpretations of the SMT draft; others were due to incomplete implementations that did not support some functions defined in the SMT draft; and still other problems could be attributed to changes in the SMT draft overlooked by some implementations. As a result of the testing, many of these problems have been fixed, and the number of interoperability problems within FDDI networks has been reduced.

Conclusion

The Common Node Software provides the FDDI project with a very flexible and stable implementation of a major portion of the FDDI data link. The initial investment in time spent on development was longer than that expected for independent software development efforts but is justified by the long-term benefits of common code. Independent development efforts for the DECconcentrator 500 and DECbridge 500 products, for example, probably would have taken less time. Each

significant bit order.

Results of Interoperability Testing

design would be based upon the hardware design and system requirements of each product. Independent designs would reduce the amount of software developed for each product because the designers

would not have to address portability issues, such as generic interfaces to the operating system or hardware.

But these two products were the first of a new network architecture, and the station management draft was not complete during product development. Thus, independent implementations would have resulted in a higher bug rate, with each product exhibiting its own set of behavior at the SMT level. Future updates and revisions to the SMT standard would have resulted in independent revisions of each product's firmware and, possibly, a new set of problems. With CNS, only one source needs to be changed and tested.

The FDDI standard promotes multivendor interoperability, so that independent products can communicate effectively in a heterogeneous FDDI network. The development of CNS significantly increased interoperability between Digital's products and those of other vendors.

The advantages of reusable software that were realized by the project team can be summarized as follows:

- The major design of interface needs to be supplied.
- Test and verification time is significantly reduced. Only the interfaces to CNS and the system dependences must be rigorously tested with the design of each new product.
- The bug rate is reduced significantly. Each new product uses the pretested and proven core library.
- The software requires little maintenance. Since the core library is stable, development only needs to be performed on the external library.

The CNS development project was the design team's introduction to reusable software. We have probably not done everything in the best possible way, but the success of the project and the time and effort saved have convinced us of the benefits of reusability.

Acknowledgments

The authors would like to thank Peter Hayden and Herman Levenson for their technical contributions to the development of CNS. Much of the structure

the software needs to be done only once. A core library and a well-defined external interface are provided. When a new product is developed, only the specific implementation of the external

of the common code is a product of their ideas. Appreciation is also extended to Bill Cronin, who was instrumental in expanding the scope and effectiveness of the interoperability testing. Finally, the authors would like to acknowledge Henry

Yang for his careful and constructive review of the paper.

References

1. FDDI Station Management, Draft Proposed American National Standard, X3T9.5/94-89, REV 6.2 (May 18, 1990).
2. H. Yang, B. Spinney, and S. Towning, "FDDI Data Link Development," Digital Technical Journal, vol. 3, no. 2 (Spring 1991, this issue): 31-41.
3. J. Hutchison, C. Baldwin, and B. Thompson, "Development of the FDDI Physical Layer," Digital Technical Journal, vol. 3, no. 2 (Spring 1991, this issue): 19-30.
4. B. Sweet, "DECelms-Managing Digital's FDDI and Ethernet Extended LAN," Digital Technical Journal, vol. 3, no. 2 (Spring 1991, this issue): 76-84.
5. D. Benson, P. Ciarfella, and P. Hayden, "FDDI-Meeting the Interoperability Challenge," Proceedings of the IEEE Fifteenth Conference on Local Computer Networks (October 1990).

Copyright 1991 Digital Equipment Corporation. Forwarding and copying of this article is permitted for personal and educational purposes without fee provided that Digital Equipment Corporation's copyright is retained with the article and that the content is not modified. This article is not to be distributed for commercial advantage. Abstracting with credit of Digital Equipment Corporation's authorship is permitted. All rights reserved.