DOT/FAA/AR-05/54

Office of Aviation Research
and Development
Washington, D.C. 20591

Handbook for Ethernet-Based Aviation Databuses: Certification and Design Considerations

November 2005

Final Report

This document is available to the U.S. public through the National Technical Information Service (NTIS), Springfield, Virginia 22161.

U.S. Department of Transportation
Federal Aviation Administration

NOTICE

Technical Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
DOT/FAA/AR-05/54		

4. Title and Subtitle	5. Report Date
HANDBOOK FOR ETHERNET-BASED AVIATION DATABUSES: CERTIFICATION AND DESIGN CONSIDERATIONS	November 2005
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Yann-Hang Lee[1], Elliott Rachlin[2], and Philip A. Scandura, Jr.[2]	

9. Performing Organization Name and Address		10. Work Unit No. (TRAIS)
[1]Arizona State University Office of Research and Sponsored Projects Box 873503 Tempe, AZ 85287	[2]Honeywell Laboratories 2111 N. 19th Avenue Phoenix, AZ 85027	11. Contract or Grant No.

12. Sponsoring Agency Name and Address	13. Type of Report and Period Covered
U.S. Department of Transportation Federal Aviation Administration Office of Aviation Research and Development Washington, DC 20591	Final Report
	14. Sponsoring Agency Code AIR-120

15. Supplementary Notes

The Federal Aviation Administration Airport and Aircraft Safety R&D Division COTR was Charles Kilgore.

16. Abstract

The purpose of this Handbook is to provide the network designer and developer with some guidelines to develop an Ethernet databus framework deployable in aircraft avionics systems. The Handbook gives design rationale and requirements for the use of Ethernet-based networks in the avionics environment and identifies the relevant issues and concerns regarding the determinism of the databus system.

The Handbook will aid in the process of qualifying an Ethernet-based databus as part of the overall aircraft certification. It focuses on identifying any and all aspects of the product that may impact its qualification. Some qualification issues related with Ethernet-based aviation databuses are discussed. The general acceptance criteria for the qualification of avionics databuses as well as the evaluation criteria specific to Ethernet-based databuses are discussed. The Handbook describes the safety, performance, and reliability requirements of an Ethernet-based databus. Using the requirements of Ethernet-based databuses as a basis, the guidelines to design Ethernet-based aviation databuses and to address nondeterministic factors are illustrated.

This Handbook does not constitute Federal Aviation Administration certification policy or guidance but may be used as input to future policy and guidance.

17. Key Words	18. Distribution Statement
Ethernet, Databus, Certification, Design, Determinism, Network, Criteria, Communication, Avionics	This document is available to the public through the National Technical Information Service (NTIS) Springfield, Virginia 22161.

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	55	

TABLE OF CONTENTS

		Page
EXECUTIVE SUMMARY		vii
1.	INTRODUCTION	1-1
1.1	Background	1-1
1.2	Purpose	1-1
1.3	Scope	1-1
1.4	Organization	1-3
1.5	Focus for Readers	1-3
1.5.1	Readers Concerned With Ethernet Databus Qualification	1-3
1.5.2	Readers Concerned With Ethernet Databus Development	1-3
2.	ETHERNET-BASED AVIATION DATABUS SYSTEM	2-1
2.1	Brief Overview of Ethernet	2-1
2.2	General Concerns of Ethernet as an Avionics Databus	2-1
3.	CERTIFICATION CONSIDERATIONS FOR ETHERNET-BASED AVIONICS DATABUSES	3-1
3.1	Background	3-1
3.2	Certification Considerations	3-1
3.3	Certification Authorities Position Paper	3-2
3.4	Generic Evaluation Criteria	3-2
3.5	Ethernet Databus-Specific Evaluation Criteria	3-3
3.5.1	Safety	3-3
3.5.2	Data Integrity	3-4
3.5.3	Performance	3-5
3.5.4	System Configuration Management	3-6
3.6	Use of COTS Products	3-7
4.	AVIONICS APPLICATION REQUIREMENTS	4-1
4.1	Determinism in a Communication System	4-1
4.2	Avionics Application Requirements	4-1
4.2.1	Traffic Pattern	4-1
4.2.2	Resource Availability	4-2
4.2.3	Bounded Service Times at End-Systems	4-2
4.2.4	Guaranteed QoS Parameters	4-2

4.2.5 Error Notification 4-3
4.2.6 Protection 4-3
4.2.7 Standard Interface 4-3
4.2.8 System Configuration 4-4

5. DESIGN CONSIDERATIONS FOR MAKING ETHERNET-BASED AVIATION DATABUSES 5-1

5.1 The CSMA/CD Protocol 5-1

5.1.1 Ethernet Bus-Based Network Topology 5-1
5.1.2 Ethernet Switch-Based Network Topology 5-3

5.2 Flow Control 5-3

5.2.1 Flow Specification 5-4
5.2.2 Traffic Shaping (Smoothing) and Policing 5-4
5.2.3 Call Admission Control 5-6

5.3 Deterministic Message Transmission in Switched Network 5-7

5.3.1 Switch Architectures 5-7
5.3.2 Performance-Guaranteed Service Disciplines 5-9
5.3.3 Multicast Traffic Support in Switch Architectures 5-11

5.4 Data Integrity and Reliability 5-13

5.4.1 System-Level Reliability 5-13
5.4.2 Message-Level Reliability 5-14

5.5 Network Stack Processing 5-15

5.5.1 Use of Connection-Oriented or Connection-Less Protocols 5-15
5.5.2 Address Resolution Mechanisms 5-17
5.5.3 Networking Address Modes 5-18

5.6 Nondeterminism in Hardware Components Within End-System 5-18
5.7 System Configuration 5-19

6. SUMMARY 6-1

7. REFERENCES 7-1

8. GLOSSARY OF TERMS 8-1

APPENDIX A—NONDETERMINISTIC OPERATIONS OF ETHERNET NETWORK INTERFACE CARD ON A PCI BUS

LIST OF FIGURES

Figure		Page
1	Scope of the Handbook	1-2

LIST OF TABLES

Table		Page
3-1	Generic vs Application-Specific Databus Evaluation Criteria	3-3

LIST OF ACRONYMS

ADN	Aircraft Data Network
ARP	Address resolution protocol
CAC	Call admission controller
CAST	Certification Authorities Software Team
COTS	Commercial off-the-shelf
CPU	Central processing unit
CSMA/CD	Carrier sense multiple access and collision detection
D-EDD	Delay earliest due data
DMA	Direct memory access
EC	Evaluation criteria
FAA	Federal Aviation Administration.
FCFS	First-come, first-served
FIFO	First-in, first-out
GPS	Generalized processor sharing
HOL	Head of line
IEEE	Institute of Electrical and Electronic Engineers
I/O	Input/output
IP	Internet protocol
IQ	Input queued
ISR	Interrupt service routine
MAC	Media access control
Mbps	Megabits per second
OSI	Open System Interconnection
OQ	Output queued
PCI	Peripheral Component Interconnect
QoS	Quality of service
RCS	Rate-controlled services
RFC	Request for comment
RR	Round robin

EXECUTIVE SUMMARY

Since safety-critical, real-time systems require deterministic behaviors in their communication operations, specialized communication networks have been used in the areas of avionics, defense, and space applications. With the advent of higher-performance computing and communication systems, aircraft will have the capability to process an unprecedented amount of information pertaining to performance, safety, and efficiency. Flight instruments will be integrated to share information and to cooperate with each other. It is inevitable that a high-speed and versatile network infrastructure will be required in the next generation of aircraft.

One commercial off-the-shelf technology, Ethernet, is seen as potentially attractive in aircraft avionics systems, due to high bandwidth, low wire counts, and low cost. It has been used in several aviation subsystems and is being considered to extend to flight-critical avionics systems where the deterministic operations are required.

This Handbook is intended to provide the network designer and developer with some guidelines to develop an Ethernet-based databus framework deployable in certifiable avionics systems. The Handbook gives design requirements for the use of Ethernet-based networks in the avionics environment and identifies the relevant issues and concerns regarding the determinism of the system and certification considerations. The handbook will aid in the process of qualifying an Ethernet-based databus as part of the overall aircraft certification.

The Handbook was developed with Federal Aviation Administration (FAA) Certification Authorities Software Team Position Paper 16 as the basis. The Handbook discusses the safety, performance, and reliability requirements of an Ethernet-based databus. The Handbook also discusses the design and certification considerations by elaborating what should be addressed to meet the system requirements. When the acceptance criteria are discussed, it focuses more on the application-specific criteria (for databuses) instead of the general criteria for design process required by RTCA DO-178B. Thus, the Handbook focuses on two issues for Ethernet databuses: (1) what should be achieved as an avionics equipment and (2) what should be addressed to meet the specification and requirements.

For avionics databus operations, this Handbook defines determinism for a communication system and identifies the nondeterministic factors that must be resolved before deploying Ethernet in aircraft networks. It also describes the communication requirements for the whole avionics application. These requirements define the performance, safety, and reliability goals that must be realized by the targeted Ethernet-based databuses. The certifiability can then be accomplished only when the required characteristics can be demonstrated by proof and systematic testing. To illustrate how an Ethernet-based databus can provide guaranteed services, the design issues from hardware interfaces to network stack software and switch architecture are discussed.

The material contained in this Handbook is part of the work done for the research project titled “Safety and Certification Approaches for Ethernet-Based Aviation Databuses,” and was funded by the FAA.

1. INTRODUCTION.

1.1 BACKGROUND.

Since safety-critical, real-time systems require deterministic behavior in their communication operations, specialized communication networks have been used in the areas of aviation, defense, and space applications. These type of specialized communication network are usually expensive, have a nominal bandwidth and complex wiring configurations, and have limited engineering support. With the advent of higher-performance computing and communication systems, aircraft will have the capability to process an unprecedented amount of information pertaining to performance, safety, and efficiency. Flight instruments will be integrated to share information and to cooperate with each other. It is inevitable that a high-speed and versatile network infrastructure will be required in the next generation of aircraft.

One commercial off-the-shelf (COTS) technology, Ethernet, is seen as potentially attractive in avionics systems, due to its high bandwidth, low wire counts, and low cost. To date, Ethernet has been used in some non-flight-critical applications and minimally in critical applications. However, the industry plans to use it more extensively and in more critical systems in the near future. There are several safety concerns when Ethernet is applied to flight-critical systems, including traffic control and quality of service (QoS) guarantee. Significant technical considerations and design issues should be examined before Ethernet can be a viable candidate as an aviation databus technology.

1.2 PURPOSE.

The Handbook is intended to provide the network designer and developer with some guidelines to develop an Ethernet-based databus framework deployable in certifiable avionics systems. The Handbook gives design rationale and requirements for the use of Ethernet-based networks in the avionics environment and identifies the relevant issues and concerns regarding the determinism of the system.

This Handbook takes the position that qualification of an Ethernet-based databus may be accomplished only when safety, data integrity, performance, and system configuration management are demonstrated sufficiently as determined by the local aircraft certification specialist using the evaluation criteria discussed in section 3.5. If Ethernet is deployed as avionics databus, then it cannot introduce any risk to the aircraft. Note that this is different from the notion of the product adding quality to the system. An avionics subsystem like the Ethernet databus may indeed do that, but this Handbook does not deal with the addition of quality. Rather, it focuses on identifying and preventing any and all aspects of the product that may impact its qualification. The Handbook describes the safety, performance, and reliability requirements of an Ethernet-based databus.

1.3 SCOPE.

This Handbook provides guidelines to use Ethernet-based databuses in a certifiable avionics network. It is intended to be informal and educational. This Handbook does not focus on any specific solution, but addresses the concerns known to date related to the use of Ethernet as an

avionics databus. It is not being used as a standalone product, but rather as input when considering issues in a project-specific context.

The intended time frame for use of this Handbook spans the entire life cycle of the Ethernet-based system under consideration. Specifically, the document is intended to be used initially during the early concept and design phase of the avionics system and then in all later phases leading up to the actual aircraft certification itself. It is intended that the use of this document throughout the entire life cycle of the development process will result in a very transparent process with minimal misunderstandings.

The Handbook was developed with the Federal Aviation Administration (FAA) Certification Authorities Software Team Position Paper 16 (CAST)-16 [1] as the basis. The design and certification considerations are discussed through elaborations on what should be addressed to meet the system requirements. Figure 1 encompasses the entire scope of the Handbook. When the acceptance criteria are considered, evaluation certification items are identified, and the focus is more on the application-specific criteria (for databuses) instead of the general ones such as those shown in DO-178B. When system and application areas are considered, functional and system configuration items are identified. Then both the evaluation criteria and functional/system items are combined to give the total Ethernet Databus Considerations. These Ethernet Databus Considerations can then be first evaluated for (1) what specification requirements should be achieved as an avionics equipment and then they can be evaluated for (2) what design issues should be addressed to meet the specification requirements.

Handbook Scope

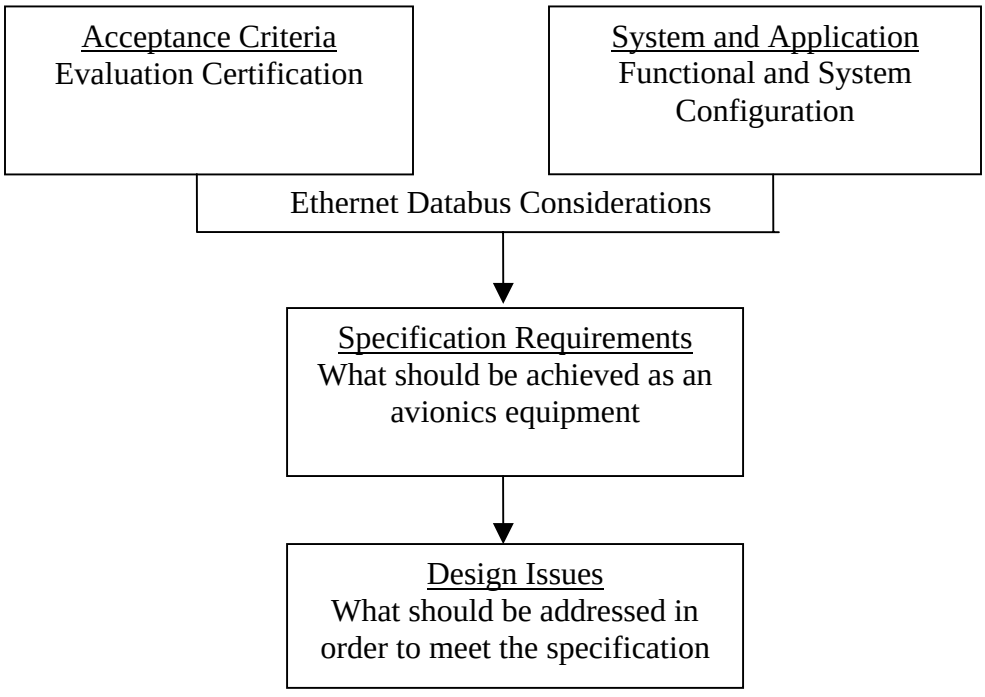

FIGURE 1. SCOPE OF THE HANDBOOK

The material contained in this Handbook is part of the work done for the research project titled "Safety and Certification Approaches for Ethernet-Based Aviation Databuses" [2].

It must be noted that this Handbook does not constitute FAA policy or guidance; rather, it is the result of FAA-funded research, which may contribute to future policy or guidance.

1.4 ORGANIZATION.

In this Handbook, section 1 provides introductory material, including purpose, scope, document structure, and suggested audience. Section 2 provides an overview of Ethernet-based aviation databus. Section 3 identifies the certification issues related with Ethernet-based aviation databus. It describes general acceptance criteria for the qualification of future avionics databuses. Finally, it addresses the evaluation criteria specific to Ethernet databus. Section 4 describes the determinism in a communication system and identifies avionics application requirements that should be satisfied by the Ethernet databus solution. Section 5 provides the design considerations for Ethernet-based aviation databuses. It addresses nondeterministic factors in the Ethernet-based databus system and provides recommendations for the deterministic design. Section 6 concludes the Handbook, while section 7 provides a list of references. Finally, section 8 provides a glossary of key terms used in this Handbook.

1.5 FOCUS FOR READERS.

This Handbook provides information for a variety of readers. Two of the targeted reader groups are described below, with a suggested approach for reading this Handbook.

1.5.1 Readers Concerned With Ethernet Databus Qualification.

Readers concerned about the certification process should understand the certification considerations of the aviation databus. The general acceptance criteria and, more specifically, acceptance criteria defined for the Ethernet-based aviation databus, described in section 3, will help such readers to evaluate the databus for avionics requirements. The avionics application requirements listed in section 4 will help these readers to validate the databus solution to be qualified. Browsing through section 5 will also be helpful for these readers to understand the concerns in deploying Ethernet in avionics environment.

1.5.2 Readers Concerned With Ethernet Databus Development.

The system architects and designers, developing a solution for an Ethernet-based, real-time communication framework for an avionics domain, should read sections 4 and 5 to understand the nondeterministic factors involved in an Ethernet-based communication framework and various design concerns to achieve end-to-end determinism in an Ethernet-based avionics system. To facilitate the certification of a system using an Ethernet databus, the designers and developers should be aware of the certification issues related to evaluation criteria for a generic avionics databus and an Ethernet-specific databus. Details in section 3 will help to understand these certification issues.

2. ETHERNET-BASED AVIATION DATABUS SYSTEM.

This section provides the overview of using an Ethernet databus and why it is preferred in some avionics environments. It also introduces basic concerns associated in deploying Ethernet onboard an aircraft.

2.1 BRIEF OVERVIEW OF ETHERNET.

Ethernet was developed at Xerox in 1973 as a part of a research program on personal workstations and distributed systems. Ethernet is a simple or branching bus-like network that uses low-loss coaxial cable linked by repeaters. It has a carrier sense multiple access with collision detection (CSMA/CD) network classification. The most common way of carrying out the CSMA/CD communications is by broadcasting packets of data on a cable that is accessible to all of the stations on the network. All stations are continually listening to the cable for packets that are addressed to them. To avoid packet collision, Ethernet implements three mechanisms: (1) carrier sensing, (2) collision detection, and (3) back off. In carrier sensing, the interface hardware at each station listens for the existence of a signal in the cable. For collision detection, the sending station also listens on its input port to make sure that the message sent and received is the same. Back off occurs when a collision is detected and a jamming signal is sent. Back off is the process of waiting for a period of time before retransmitting the message.

The standard for Ethernet is Institute of Electrical and Electronic Engineers (IEEE) 802.3. There have been three major standardization efforts with the IEEE 802.3 standard. The 1990 version operated at a maximum of 10 megabits per second (Mbps) and is commonly known as 10BaseT. The 1995 version operated at a maximum of 100 Mbps and is commonly referred to as 100BaseT. In 1998, the Gigabit Ethernet was standardized. It operates at a maximum of 1 gigabit per second. The Gigabit Ethernet is compatible with the slower Ethernets.

2.2 GENERAL CONCERNS OF ETHERNET AS AN AVIONICS DATABUS.

The advancements in data processing of avionics applications have been resulting in higher bandwidth requirements. Aviation instruments are also moving from the use of low bandwidth data communication to high bandwidth. Next-generation aircraft warning systems, for example, may want to combine map and air traffic data, terrain information, weather radar returns, information on man-made obstacles, and imagery on the airport environment, and finally, fuse this information into a single three-dimensional representation. The need for speedier communication to handle future growth requirements of avionic applications, such as the automatic dependent surveillance-broadcast and Future Air Navigation System (FANS), warrant a move towards the use of a high-bandwidth databus as the communication medium in aircraft [3]. Ethernet promises to accommodate future systems' bandwidth demands, increase flexibility in avionics design and procurement, and reduce aircraft wire counts, thus lowering aircraft weight and operating costs. The use of Ethernet technology translates into greater flexibility in avionics architecture design and lower aircraft construction and operational costs. Apart from bandwidth improvements, another attraction of Ethernet is its promise to free avionics data transfer in integrated avionics systems from the logistical limitations of a backplane bus [4] of the special-purpose communications bus, linking computing modules inside of a physical

enclosure. So in the future, Ethernet will likely replace the backplane bus for at least some aircraft.

An aviation databus is defined as a databus between two or more avionics instruments comprising an aircraft network. The term aircraft network is used here to indicate that it is a network of avionics instruments carrying time-critical aircraft data and, therefore, to emphasize the fact that its requirements are considerably different compared to that of a general-purpose communication network used in a nonaircraft system. An Ethernet-based aviation databus system is a system in which Ethernet is used as the interconnection medium in the aircraft network.

An avionics Ethernet-based databus needs to guarantee and control bandwidth allocation to applications on the network and ensure that the correct data arrives at the correct destination at the correct time. There will be some extra electrical requirements on the cable that also need to be considered to support the higher frequencies [5]. There are many safety concerns, however, when Ethernet is applied to flight-critical systems. Ethernet is not designed to carry time-critical data. It is inherently nondeterministic in nature, and this is a factor that undermines its use in a time-critical system such as an avionics network. The nondeterminism in Ethernet is due to its use of CSMA/CD protocol for resolving bus contentions. Even in the presence of a collision-free environment for data transfer, one cannot guarantee that a data packet will be delivered on time. This is due to the fact that traffic in a real-time system tends to be bursty at times, and this burstiness can lead to buffer overflows both at the source (the transmit and receive buffers maintained by the Ethernet controller) and at the intermediate nodes, leading to packet loss. Therefore, to support avionics applications, the standard Ethernet will have to be supplemented with additional solutions to overcome this safety concern of nondeterminism, and the QoS issues in the whole aviation network, including network configuration, message routing, and scheduling, must be considered.

As an industrywide effort, ARINC's AEEC has established a working group on Aircraft Data Network (ADN). The ARINC Specification 664 defines an Ethernet data network for aircraft installation and is in multiple parts for the protocol operations of different layers, network addressing, and services. For deterministic operations, part 7 of ADN 664 focuses in the Avionics Full Duplex Switched Ethernet in which quality of service including timely delivery is paramount [6]. The specification adopts the virtual link concept to describe the quality of service requirements of a logical unidirectional connection between a source end-system and one or more destination end-systems. A guaranteed service on a virtual link is then required so that (1) bandwidth and bounded latency are guaranteed, and (2) a bound to the delay jitter can be mathematically computed.

In addition to the development of a deterministic solution, Ethernet databus designers should consider the safety and certification requirement of avionics equipment. As with any hardware and software components, the safety process of using Ethernet as an aviation databus begins with an aircraft-level assessment of functions and hazards (using the SAE ARP4761 approach [7]). The guidelines for development, including DO-254 [8] for complex hardware and DO-178B [9] for software, should be followed. In addition, several fundamental issues, including scheduling, safety, and fault-tolerance, should be examined carefully since Ethernet networks are designed

for commercial non-safety-critical applications. The certification issues and design guidelines to address the safety, performance, and reliability of an Ethernet databus are described in following sections.

Also, an Ethernet-based aviation databus should be verified to ensure that the avionics application requirements are being satisfied. The system should be tested in the practical environment using real airborne instruments. During the development phase, data traffic may be produced through a workload generator that synthesizes traffic load profiles from avionics applications. By using the communication specification of real-life avionics applications, a profile of message characteristics can be established. This approach allows scaling the communication requirements and adjusting the traffic loads based on anticipated applications. The synthesized workload can then be supplied to Ethernet-based aviation databuses under various workload parameters.

3. CERTIFICATION CONSIDERATIONS FOR ETHERNET-BASED AVIONICS DATABUSES.

This section describes the certification considerations that should be addressed when using databus technology in avionics systems. It describes the general evaluation criteria for avionics databuses and the evaluation criteria specific to Ethernet-based avionics databuses. Understanding the evaluation criteria and applying them throughout databus product design, implementation, and testing life cycle will assist designers in the development of a certifiable solution.

3.1 BACKGROUND.

With respect to the use of Ethernet-based avionics networks, it is desirable to create a framework for collaboration between the certification authority and the applicant/developer to be used in support of the certification process of an aircraft and its systems. Such a framework should address methods by which the goals of safety, performance, and reliability of an Ethernet-based avionics subsystem can be addressed. Because this will vary from system to system, this framework necessarily describes a generic approach. Ultimately, however, the applicant/developer should create a product-specific version that specifically calls out the techniques to be employed in addressing the relevant issues to their system implementation.

3.2 CERTIFICATION CONSIDERATIONS.

An avionics databus cannot be certified independently; rather, it must be qualified as part of an aircraft system or function, that is, under the type certificate process [10]. The complete certification cycle includes type certification (design approval), production certification (production approval), airworthiness certification (airworthiness approval), and continued airworthiness management (support for the life of the aircraft). The assertion that an avionics (or other) product is certifiable requires the individual product to meet the same level of regulatory and safety criteria applicable to the system in which it is installed.

In general, certifiability is accomplished only when safety, performance, and reliability can be demonstrated to comply with applicable regulatory requirements. This may occur through any number of methods, including, but not limited to, inductive proof, extensive testing, code reviews, and coding metrics (code coverage, branch analysis, etc). Two key standards that address design assurance methods are DO-178B [9] and DO-254 [8] (for software and hardware, respectively). The concept of design assurance refers to the level of process rigor applied to the design and building of a product, such that one can provide adequate confidence and evidence that a product satisfies its given set of requirements. In general, these documents provide guidelines for the production of software and hardware for airborne systems and equipment, which performs its intended function with a level of confidence in safety and complies with airworthiness requirements.

The applicant/developer must specify at the outset exactly how they will ultimately demonstrate the achievement of safety, performance, and reliability. These specifications are provided in the form of a product-specific certification plan or other equivalent certification document [11]. The applicant submits this to the certification authority for consideration, and after one or more

rounds of feedback and enhancement, the document becomes part of the product certification document set. The certification document remains with the product throughout the entire development and certification process, guiding the developers in their work and keeping the certification authority informed of any shortcomings or issues that may arise. If the processes described in the document are followed throughout product development, there should be no certification-related requirements surprises at the end of the development activity.

3.3 CERTIFICATION AUTHORITIES POSITION PAPER.

In early 2003, the international CAST published Position Paper (CAST-16) "Databus Evaluation Criteria," applicable to any databus technology proposed for aircraft use[1]. As stated in the abstract:

> "A number of new and existing databuses are being proposed for use by aircraft manufacturers. This paper documents criteria that should be considered by databus manufacturers, aircraft applicants, and certification authorities when developing, selecting, integrating, or approving a databus technology in the context of an aircraft project". [1]

The paper addresses eight major categories that should be considered when evaluating a specific databus technology. Within each category, specific criteria are enumerated[2]. It should be noted that the categories and criteria provide a minimum listing, and each specific databus architecture should be evaluated to address any additional categories or criteria that may be applicable. The eight categories and number of criteria in each are:

1. Safety—9 criteria
2. Data Integrity—12 criteria
3. Performance—10 criteria
4. Design/Development Assurance—2 criteria
5. Electromagnetic Compatibility—4 criteria
6. Verification and Validation—9 criteria
7. System Configuration Management—5 criteria
8. Continued Airworthiness—1 criterion

3.4 GENERIC EVALUATION CRITERIA.

Of the eight major categories and specific criteria listed in section 3.3, many are generic in nature and must be evaluated regardless of the specific databus technology selected. The remaining criteria are application- and/or implementation-specific and can only be evaluated in the context of a specific databus technology and network topology. Table 3-1 differentiates between these

[1] It is important to note that all CAST papers include the following disclaimer: "This document is provided for educational and informational purposes only and should be discussed with the appropriate certification authority when considering for actual projects. It does not constitute official policy or guidance from any of the authorities."

[2] The reader should note that some categories overlap, consequently evaluation criteria may appear more than once within the CAST paper.

generic and application-specific criteria. In subsequent sections of this handbook, the application-specific criteria are discussed for Ethernet. The generic criteria are not discussed, since they apply to all databuses. However, readers are encouraged to review the CAST paper for themselves.

TABLE 3-1. GENERIC VS APPLICATION-SPECIFIC DATABUS EVALUATION CRITERIA

Category Number	Category	Criteria	Generic or Application-Specific
1	Safety	1, 2	Generic
		3…9	Application-specific
2	Data Integrity	All (12)	Application-specific
3	Performance	All (10)	Application-specific
4	Design/Development Assurance	All (2)	Generic
5	Electromagnetic Compatibility	All (4)	Generic
6	Verification and Validation	All (9)	Generic
7	System Configuration Management	All (5)	Application-specific
8	Continued Airworthiness	All (1)	Generic

3.5 ETHERNET DATABUS-SPECIFIC EVALUATION CRITERIA.

The general evaluation criteria, identified in the previous sections, must be considered for any aviation databus that will be used in an airborne system. This section describes evaluation criteria specific to an Ethernet-based aviation databus and must be considered in addition to the general evaluation criteria.

Ethernet-specific evaluation criteria focus on application-specific categories of the general evaluation criteria described above. As such, only the following categories from table 3-1 are explained.

- 1—Safety
- 2—Data Integrity
- 3—Performance
- 7—System Configuration Management

3.5.1 Safety.

The safety evaluation criteria for a databus are addressed as the potential incompatibility between various hardware and software components of the databus architecture, resulting from the use of COTS versions of those components. As the COTS components are not designed following the safety and certification considerations, they induce most of these incompatibility issues. Since Ethernet was essentially designed for a general-purpose communication network, such as the internal network of an organization, it has certain drawbacks when it is used for time-

critical communication that is mainly due to its inherent nondeterminism. To ensure safety in an Ethernet-based databus system, one has to ensure that data delivered through such a system is deterministic. This implies that the end-to-end delay and other delay parameters (such as delay jitter and the latency of the data packet along each node in the network) are bounded and consistent.

The safety evaluation criteria (EC) that should be addressed as part of any solution are discussed in section 3.1 of reference 1 and are summarized below.

- EC 3: Bus availability and reliability must meet the safety requirements, determined by the safety assessment.
- EC 4: Partitioning/protection of the bus architecture, implementation, and failure detection and management features should be demonstrated.
- EC 5: Failures should be detected and managed (e.g., using redundancy, detecting loss of nodes, supporting transparent shadow nodes, and supporting replica determinism or parallel nodes).
- EC 6: Common cause (including common mode) failures should be addressed.
- EC 7: Reconfiguration of node/network should be addressed (i.e., address safety aspects of possible configurations and states), as appropriate.
- EC 8: A strategy for future bus expansion and associated effects on system integrity/safety should be developed.
- EC 9: Redundancy management (loss reporting and automation) should be addressed.

3.5.2 Data Integrity.

Data integrity refers to the degree of correctness of the data transferred between entities residing on the network. Data integrity, sometimes identified with reliability, can be considered at two levels: one at the system level and the other at the message level. At the system level, reliability refers to the ability of the system/network to withstand node and link failures. In this sense, one can think of it as the degree of fault tolerance of the network. Message level reliability guarantees that messages are delivered without any error, or any introduced error is detected with high reliability when received.

The data integrity EC that should be addressed as part of any solution are discussed in section 3.2 of reference 1 and are summarized below.

- EC 1: The maximum error rate per byte expected for data transmission should be defined.
- EC 2: A means to detect and recover from the errors should be implemented to meet the required safety assessment criteria (e.g., cyclic redundancy codes, built-in detection, hardware mechanisms, and architecture provisions).

- EC 3: A data load analysis should be performed to specify limitations of the databus.
- EC 4: Bus capacity should be defined.
- EC 5: Buffer overflow and underflow should be addressed.
- EC 6: Bus integrity issues such as babbling/jabbering devices, packet collisions, broadcast storms, and incomplete packages should be considered.
- EC 7: Reconfiguration of node/network should be addressed and shown to support data integrity requirements, if reconfiguration is enabled.
- EC 8: Bidirectional implementation should be carefully evaluated to address all data integrity issues.
- EC 9: Switch saturation should be addressed, if switches are used.
- EC 10: Mechanisms to implement reconfiguration (both software and hardware) should be evaluated and addressed for data integrity assurance, if such mechanisms are used.
- EC 11: The level of allowable degraded functionality must be considered.
- EC 12: Any security issues that may affect data integrity should be addressed.

3.5.3 Performance.

The performance of a system is one of the most important and necessary aspects of any acceptance criteria. The performance of a databus is measured in terms of the bus bandwidth capabilities, per transmission overhead, data latency, and the like. The performance measures used for an Ethernet-based aviation databus are no different than for a generic databus.

The performance EC that should be addressed as part of any solution are discussed in section 3.3 of reference 1 and are summarized below.

- EC 1: The bus operating speed and scheduling of messages (timing and prioritization) should be evaluated and shown to support the safety and integrity requirements.
- EC 2: Loss of bus, shorting of bus, or opening of bus situations should be evaluated.
- EC 3: The system interoperability (including bus topology, communication protocol, and any other aspects) should be evaluated by the applicant.
- EC 4: Bus length, stub length, and number of participant limitations should be established and followed.
- EC 5: Degraded operation/performance should be defined and validated.
- EC 6: Retry algorithms should be evaluated.

- EC 7: Bus bandwidth capabilities and limitations should be documented and adhered to.

- EC 8: Data latency and efficiency should be documented.

- EC 9: Per transmission overhead and other overhead effects should be established.

- EC 10: Failure management should be evaluated for adequacy and effects of failures within.

3.5.4 System Configuration Management.

The way a system is configured can greatly impact its performance and efficiency. Configuration parameters for a databus system may include facilities to specify the type of applications that can communicate through the databus, the number of such applications that can communicate simultaneously, the amount of bandwidth reserved for control data, and so on. Consequently, each individual aircraft's databus configuration could potentially be unique, thus requiring a formal configuration management program.

The configuration management EC that should be addressed as part of any solution are discussed in section 3.7 of reference 1 and are summarized below.

- EC 1: The integration of the databus into the aircraft design must be viewed from a total systems perspective. Therefore, mechanisms of configuration control, both from the fleet and the individual aircraft basis, must be assured. Controllability of the modifications or additions of nodes and applications on the bus is an absolute necessity to ensure continued operational safety, both in certification and field maintenance activities.

- EC 2: Configuration control is required in all phases, from design through production and maintenance, and is accomplished via use of standards and documentation procedures. Configuration control should address the overall databus and any options. Any changes of configuration would require an additional certification effort.

- EC 3: Specification standards should be documented. The following is a minimal list of appropriate specification standards needed: physical rules (e.g., transmission media, connectors, terminations, maximum number nodes/run lengths, etc.) and logical rules (e.g., message packet definitions).

- EC 4: The following documents should be provided and adhered to, as a minimum: interface control documents, designer's guide, and installer's guide.

  EC 5: If the databus uses a multilayered architecture (e.g., the Open Systems Interconnection (OSI) standard: Physical (OSI Layer 1), Datalink (OSI Layer 2), Network (OSI Layer 3), Transport (OSI Layer 4), Session (OSI Layer 5), Presentation (OSI Layer 6), Application (OSI Layer 7)), documentation to support all layers should be provided.

3.6 USE OF COTS PRODUCTS.

Much of today's modern databus technology is based upon COTS products. While not explicitly mentioned in the CAST paper, several issues associated with the use of COTS must be addressed:

- Product licensing, royalties, and data rights (primarily a business issue)
- Availability of databus artifacts in support of hardware and software design assurance (e.g., requirements documents, verification results, review records, etc.)
- Suitability of databus hardware and software for avionics environment (e.g., high vibration, radiated emissions, lighting tolerance, etc.)
- Obsolescence support and continued airworthiness (COTS product families are continually improved and retired, whereas aircraft products must support 20+ year lifespans)
- Databus security (especially in wireless networks and those that connect to passenger entertainment)

Addressing these issues requires coordination and cooperation between the databus supplier, system integrator, applicant, and certification authority (e.g., FAA).

4. AVIONICS APPLICATION REQUIREMENTS.

To satisfy the Ethernet-based avionics databus-specific evaluation criteria, application requirements should be addressed. This section describes the determinism in a communication system and lists the avionics application requirements that should be satisfied by any databus framework.

4.1 DETERMINISM IN A COMMUNICATION SYSTEM.

Before listing the details of avionics application requirements, let us define clearly what is meant by determinism in a communication system. Determinism, at a network level, can be defined as the ability of the network to guarantee delivery of messages to the nodes belonging to that network within a specified period of time. At the individual node level, the system is deterministic if it processes an input/output message in a time-predictable fashion. Determinism at both levels is equally important and together they lead to a highly dependable system, where the nodes represent the various avionics applications with the Ethernet acting as the medium of communication between them. In fact, in this scenario, the two forms of determinism are very strongly intertwined and each drives the other.

The above definition is limited to the cases in which the systems have no failures. In fact, with possible random failures (no matter how reliable the systems are), the systems can be recovered and be operated in a degraded mode. The message delivery can still be guaranteed, but could have different bounds comparing with the failure-free systems.

There are many factors affecting the deterministic transmission/reception of data on Ethernet. They are related to how the networks are operated and the amount of traffic transmitted by each node. A summary of the factors that should affect all systems is given below.

- The amount of bandwidth available on the network.
- The number of nodes connected to the network.
- The average rate of traffic flow through the network and its burstiness.
- The protocols used for data communication and conflict resolution on the network.
- The interconnection hardware and the network stack software.

4.2 AVIONICS APPLICATION REQUIREMENTS.

To support the avionics applications, the databus designers should understand the avionics application requirements from the underlying framework.

4.2.1 Traffic Pattern.

As a databus-enabling communication system in which avionic applications exchange messages for their computations, the traffic patterns appearing on the databus should be specified. The traffic patterns will be viewed first at the application level, i.e., a source application is the message transmitter and one or more destination applications are the receiver. The messages may be sent in a periodic manner as the source application is invoked periodically, or they can be triggered by aperiodic events. As applications are assigned to run in end-systems, messages are

routed from source nodes to destination nodes according to the network topology. Thus, the traffic patterns in the network can be described by

- the connections where a sequence of messages will be transmitted by the source applications and received at the destination applications.

- the message arrival process and the length of messages for each connection from a source application to a set of destination applications.

- the routing strategy and network topology.

4.2.2 Resource Availability.

The underlying communication subsystem should guarantee the resource requirements of applications. During the connection setup, the application can describe all the resource requirements. The requirements can include the bandwidth needed for each connection and the buffer space that is used to temporarily save blocked messages at end-systems and in the switches. The system should analyze these requirements such that the number of available system resources should be able to meet the application-specific requirements. If connection is accepted, then the allocated resources to that session should be available as long as the session is active.

4.2.3 Bounded Service Times at End-Systems.

The service times of network stack software at end-systems must be bounded. That is, application programming interface calls made by the application for establishing or releasing a session and sending and receiving data for an active session should have a predetermined worst-case service time. After this time, the call should return to the application with either a success or failure. Calls made by the administrative application for configuration purposes (e.g., adding a route entry in the routing table) should also have bounded service times. The service time bounds should be considered when the application tasks are analyzed for schedulability.

4.2.4 Guaranteed QoS Parameters.

Each avionics application should specify its service requirements, such as the required transmission rate, traffic burst size, message delay, and jitter, through a well-defined flow specification. The underlying communication subsystem should guarantee that these QoS parameters are satisfied over the period during which the session is active.

The other QoS parameter is the probability of packet dropping. Packets may be dropped due to insufficient buffer space, bit errors in transmission, or hardware and software failures. If a message is lost or contains errors, the erroneous condition should be detected within a bounded detection latency.

4.2.5 Error Notification.

The application may require receiving the error notifications related to its connection flow. These errors can be categorized as errors within the end-system and errors within a network. The source of errors can be random hardware failure. On the other hand, errors can be caused by software or design faults (one cannot guarantee that the system is error-free). In either case, the avionics application should be notified of the errors within the end-system. However, whether the sender application should be notified of errors occurring in the network, e.g., packet drops, or in the receiver end-system, e.g., buffer overflow, is an open question to be decided on an application-by-application basis. These notifications might be necessary, depending upon the type of traffic or criticality of the error. However, these notifications lead to added per transmission overhead, network congestion, and worst-case execution time effects.

4.2.6 Protection.

An application can expect its data to be protected from corruption during the transmission on the network or due to interference from other applications in the end-system.

The communication subsystem should guarantee that the data will never be corrupted and/or that error detection/recovery mechanisms are in place. Also, the system should ensure proper protection among application partitions such that the resources allocated to one partition are never accessed by the applications of other partitions and that an application's data can be read only by its intended recipient.

Faults occurring in the system by an application or misbehaving of an application should never affect other applications by any means. For instance, one of the applications may violate its flow specification and try to send more data than specified transmission rate. This violation should not affect the QoS of other applications. The databus framework should guarantee the strong fault isolation by following strong partitioning of the system resources and channel allocations of the end-system and the whole network [12].

The session configuration interfaces are available for each application, but the system configuration interfaces should be available only to special-purpose applications such as network administration functions. The underlying framework should define the access capabilities, guaranteeing that the whole system will not be affected by the nonadministrative or malicious application.

4.2.7 Standard Interface.

There should be a standard, well-defined interface between the applications and the communication subsystem. These interfaces should be descriptive and broad enough so that the application can mention its service requirements, resource requirements, configuration parameters of the system, etc. The system developers can follow interface semantics similar to socket interfaces or device semantics or any standard application programming interfaces. The system configuration parameters should have the same semantics and should be standardized, e.g., the addressing formats like Internet Protocol (IP) addressing for all nodes should be defined.

4.2.8 System Configuration.

Ethernet-based databuses will enable communication mechanisms between various applications. Thus, the databus systems should be configured according to the target applications and their requirements. The factors to be considered in a system configuration include network topology, node addressing scheme, resource allocation, etc. In addition, the system should be flexible enough to be reconfigured or changed dynamically, e.g., an application may have the facility to reconfigure its session parameters. This dynamic reconfiguration of the session may induce nondeterminism in the end-system. Hence, it is required that the underlying communication subsystem should provide the guarantee of deterministic behavior of the system, if it is supporting session reconfiguration. To be flexible, addressing should be at an abstract level, e.g., multicast address/IP address, or better still, machine name-based addressing.

5. DESIGN CONSIDERATIONS FOR MAKING ETHERNET-BASED AVIATION DATABUSES.

This section provides design considerations for Ethernet-based databus as by describing various factors contributing towards the determinism of the system. It also identifies the factors contributing to nondeterminism in the nodes, as well as in the Ethernet medium used by the nodes for communication.

Below is a set of issues that should be addressed, at a minimum, by the databus designers to make Ethernet suitable for real-time communication. It will also provide some design recommendations to consider while addressing these issues.

5.1 THE CSMA/CD PROTOCOL.

Ethernet is an IEEE 802.3 standard that uses CSMA/CD mechanism to resolve contention on the network in case of simultaneous data transmission by multiple nodes.

As defined in the CSMA/CD protocol, a node in the network can transmit messages only if the network is idle. The node has to wait if the network is busy. If two nodes try to transmit a packet simultaneously, a collision will occur. The two transmitting nodes listen to the network to detect a message collision. If a collision is detected, the transmitting nodes wait a random length of time to retry transmission. The standard binary exponential back-off algorithm determines this random length of time. The randomness of this waiting time essentially makes CSMA/CD nondeterministic.

To avoid the collision problem and to attain deterministic message communication, the network topology of the underlying communication subsystem plays an important role. The determinism in CSMA/CD can be achieved with the network topology, which assures that the collision will not occur on the network at any time. Two approaches can be considered for Ethernet databus developers:

- Ethernet bus-based network topology
- Ethernet switch-based network topology

5.1.1 Ethernet Bus-Based Network Topology.

Deterministic message transmission and collision prevention can be achieved through an Ethernet bus-based configuration by following the time-based protocol mechanisms. Various time-based protocols can be implemented in the system, depending on the system requirements.

If the whole system can be configured statically, then off-line schedule determining the time slots for all nodes can be generated. At run time, this schedule should be followed by all nodes in the system. In any given time slot, transmission of data by only one node guarantees that there will not be any collisions in the system.

In another approach, token passing protocols can be deployed at the data link layer such that only the node having a token at a particular instant of time can only send data on the network. This token is circulated to all nodes sequentially so that each node will get a chance to transmit.

These kinds of time-based approaches can be deployed successfully in Ethernet-based systems to avoid collisions. Though these kinds of schemes differ in their functionality, they are based on a basic concept of time slice reservation and synchronized operations.

Ethernet databus designers should address standard issues related to the time slice reservation.

- Babbling idiot. The system should make sure that one of the nodes should not monopolize the bus and, in turn, the network bandwidth. If the node does not relinquish control of the bus by itself, the system should provide a way to recover from that situation.

- Time synchronization. These protocols are strictly based on time. Thus, in order to follow the schedule of network bus allocation, the clocks of all nodes should be synchronized.

- Different speeds of different hardware. The hardware configuration of nodes might differ from each other, e.g., central processing unit (CPU) speed. For correct time measurements, all CPU speeds should be known at design time and corresponding time allocation, for each node should be corrected to ensure fairness.

- Fault isolation. The system should ensure that it can recover from any faults such as the loss of tokens, multiple tokens, or babbling idiot nodes.

- QoS handling/prioritizing real-time and non-real-time traffic. All time-based protocols should prioritize the traffic as real time and nonreal time and should satisfy the QoS requirements of differentiated services.

- Configuration issues. To ensure the deterministic behavior of the system, system parameters should be known at design time and should be bounded. For instance, to provide fairness policy and guarantee the exact time slice of each node, the packet size of the system should be limited to a given bounded size. Otherwise, the effect of variation of different packet lengths should be addressed.

- Fairness. The protocols should ensure the scheduling fairness to all nodes while accessing the network bus.

- Scalability. Performance of the system should be addressed with the varying factors in the system, such as number of nodes in the network, the size of the network, and the frame length.

5.1.2 Ethernet Switch-Based Network Topology.

Instead of sharing an Ethernet bus by multiple nodes, each node can have a dedicated Ethernet connection directly to a switch node. In a full-duplex, fully switched network, separate conductors are used for sending and receiving data, and nodes can only communicate with the switch and never directly with each other. Thus, each Ethernet bus has only one transmitter, and nodes can forego the collision detection process and transmit at will. Hence, data transmission between the switch and the nodes takes place in a collision-free environment. This virtually eliminates nondeterminism due to CSMA/CD.

A full-duplex, fully switched network topology guarantees that there will not be any collisions in the system and provides complete determinism of the system. The issues related to the switched network topology are discussed in detail in section 5.3.

5.2 FLOW CONTROL.

Irregular traffic patterns can lead to system overload, network congestion, and buffer overflows at the host and intermediary switches leading to unbounded packet delays, large delay jitter, and packet losses. These issues can be handled well by using flow control models in the system.

Flow control is a mechanism that enables a source node to find a transmission rate that matches the service rates of the receiver and that of the network. These mechanisms are used to provide the guaranteed service to real-time applications and to avoid congestion in the network. Ethernet databus developers can implement flow control mechanisms at the transport layer, network layer, and data link layer.

These mechanisms can be implemented either as a closed loop or an open loop. In closed loop flow control mechanisms, the communication subsystem explicitly/implicitly monitors the available service rate. When connection is requested, it can send data with this service rate. It can use full bandwidth of the network, depending on the monitored data. However, because of the round trip delays, monitoring algorithm complexities, and nonguaranteed service rates, the closed loop flow control mechanisms are considered to be less effective to make the system deterministic, compared to the open loop mechanisms. Thus, this discussion is restricted to the open loop flow control model.

For open loop flow control mechanisms, the application running on a source node has to provide the flow specification. The call admission control mechanism checks if it can satisfy the flow specification requirements and decides whether to accept the connection or not, and the communication subsystem of the source node guarantees that the flow specification requirements are maintained until the connection is alive and ensures that it is not violated by that application or by other applications.

Open loop flow control mechanisms are best suited for applications that can describe their traffic through flow specifications and for connections that require a guaranteed service rate, QoS guarantees. Because of complete determinism, the deployment of an open loop flow control model is strongly recommended to Ethernet-based databus developers. In the following, the

essential components of open loop flow control, including flow specification, traffic shaping, policing, and admission control, are discussed.

5.2.1 Flow Specification.

In a system supporting differentiated services, the bandwidth allocation is varied for different flows. The bandwidth allocation depends on factors such as the rate of generation of traffic, the QoS level associated with the flow, static priorities associated with the application to which the flow belongs, and the characteristics of the traffic flowing through it. The flow specification defines these factors.

Traffic analysis should be done on experimental traffic data to understand the traffic characteristics. Issues with traffic analysis are discussed in section 5.2.2.

Flow specification should be defined and handled in a such a way that

- it adequately describes flow, so that communication subsystem does not reserve too little or too much resources for that connection;
- it should characterize the worst-case behavior of the traffic in that connection;
- it should be easy to describe and use for admission control; and
- flow specification should not be changed within the communication subsystem, or the communication subsystem should guarantee that this variation will be in bound and will be accounted.

An example flow specification can be done following the definitions in request for comment (RFC)-1363 [13] and RFC-2215 [14].

5.2.2 Traffic Shaping (Smoothing) and Policing.

Flow specification can be considered as a contract between the application and communication subsystem. Traffic shaping is performed by the communication subsystem to ensure this contract is never violated by the connection. During the data transfer, traffic shaping guarantees that the traffic generated by the application always follows the flow specification and, thus, the traffic generated in the whole network stays within bounds.

Guaranteeing per connection, QoS requires that each flow be allocated a fraction of the bandwidth appropriate to its service requirement. In a differentiated service system, different flows are allocated, varying fractions of the bandwidth, depending on factors such as the rate of generation of traffic, the QoS level associated with the flow, and the characteristics of the traffic flowing through it. Once bandwidth is allocated, a traffic regulator associated with the flow can be used to enforce it, as long as the session is active.

The need for adaptive traffic smoothing can also arise in the integrated environment, where both real-time and non-real-time packets are transported on Ethernet. The traffic smoother (with

scheduler) should prioritize the real-time packets as per flow specification and should also give higher priorities to real-time packets over non-real-time packets to eliminate contention within each local node. The smoother, installed at each node, regulates a node's outgoing stream to maintain a certain traffic generation rate. To provide a sustainable throughput, the traffic generation rate is allowed to adapt itself to the underlying network load condition.

The databus developers can implement traffic shaping mechanisms, like leaky bucket or token bucket shaping. A leaky bucket model is basically an algorithm to limit the rate of traffic generated by a source. Its name is derived from the fact it behaves similar to a leaky bucket where drops trickle down through the leak in the bucket. It consists of a buffer where arriving packets are stored. Packets from this buffer can be transmitted at a particular rate called the leak rate. A token bucket is a slight modification of the leaky bucket model. It enforces an output pattern capable of sustaining bursts of traffic. However, the maximum sustainable burst size is limited and is equivalent to the depth of the token buffer. Here, the bucket holds tokens that are generated at the rate of one token every time unit. Each token is equivalent to a certain number of bytes of data. At any given point of time, the amount of data that can be sent is equal to the sum of the weights (in bytes) of all the tokens in the token bucket.

The choice of traffic shaping mechanism should correspond with the flow specification. That is, if traffic is characterized only by the transmission rate, then the leaky bucket model can be considered suitable. However, if the burst size is also mentioned in the flow specification, then the traffic shaping mechanism should be flexible enough to cope with the burstiness in traffic. In this case, a token bucket model will be more appropriate than a leaky bucket model.

Traffic policing guarantees that the application always remains within the flow specification. If the application violates the flow specification, then those packets can be dropped. Traffic policing can be configured on several criteria. If the connection exceeds the maximum transmission rate, then packets can be dropped or, if connections exceed average rate plus maximum burst size, then packets can be dropped. Also, instead of dropping the packets, packets can be marked as out-of-profile packets.

With hard real-time communication requirements for avionics applications, packet dropping is not allowed. Queuing mechanism should be deployed to store the packets in transit. There should be an upper limit on the buffer requirements for queuing. These buffer requirements can be made systemwide or per connection, depending on the QoS.

Traffic shaping mechanism deployed by the databus developers should be characterized by the following factors.

- Traffic shaping delay
- Accept rate/packet outgoing rate in the network, depending on the network capacity
- Effect of flow violation by one connection on other connections
- Maintenance of flow specification over the lifetime of connection

- Guarantee that if connection does not violate the flow specification, there should not be any out-of-profile packets or packet loss for that connection

- Buffer requirements for queuing

- Number of out-of-profile packets per connection

- Delay jitter induced because of queuing

- Packet dropping rate per connection

- Total bandwidth utilization achieved in the whole system

- Performance of traffic shaping with configuration parameters, e.g., in case of token bucket traffic shaping, the effect of shaping can be measured by varying token bucket fill rate, bucket size, size of a token, etc.

5.2.3 Call Admission Control.

A call admission controller (CAC) is a central authority through which all requests for connection establishment go. A CAC uses flow specification as the basis to compute a service rate for the flow and to perform a schedulability analysis on the flow for that service rate.

It determines whether the connection is accepted or rejected. To guarantee the service rate for a flow, a connection should be admitted only if all nodes along the path from the source to the destination can guarantee to deliver the requested QoS to the new connection without causing any deterioration of the QoS of existing connections. If no firm guarantee can be provided, then that connection should be dropped.

If the connection is accepted, it enforces QoS guarantees by allocating network and operating system resources such as transmit and receive buffers to allow the application to perform data read and write operations on the connection. Its significance is in ensuring that the network resources (such as bandwidth) are never overallocated, thus preventing system overload and overall network congestion.

An admission control can be deployed for best-effort flows, constant rate flows, or variable rate flows.

- In the case of best-effort flows, the admission controller should always accept all the connections. Thus, no service guarantee is provided with the connection.

- With constant rate flows, admission control mechanism can simply follow the rule that total bandwidth of already admitted flows and newly requested flow bandwidth should not exceed the capacity of the link.

- In the case of variable rate flows, depending on the burstiness of the traffic, the service rate is varied.

If the bandwidth is reserved by following the burstiness of traffic, then the bandwidth is never used to 100%. Otherwise, in bandwidth reservation with average transmission rate, packets will be dropped from burst. Thus, admission control should cope with average rate as well as burstiness of traffic. Traffic analysis should be carried out to categorize the avionics applications traffic so that the appropriate model of admission control can be deployed in the system.

5.3 DETERMINISTIC MESSAGE TRANSMISSION IN SWITCHED NETWORK.

Packet-based switching is an effective approach of using network bandwidth in connecting multiple nodes in switched networks. The links between an end-node and a switch node, or between switch nodes, can be a full-duplex Ethernet bus. Although the nondeterministic behavior of CSMA/CD protocol is no longer an issue (see section 5.1.2), the deterministic data transmission at each switch node is a concern. The deterministic transmission in such a switched network is mostly dependent on the switch architecture and its functionality, as well as the network traffic.

Ethernet databus developers of switched networks should make a proper choice in selecting or designing the switch architecture. The switch design should address the following issues of switch characteristics, at a minimum:

- Is there a traffic model for message arrivals at each switch node?
- Is the network configured such that all links are not overloaded under both unicast and multicast traffic flows?
- Does the message scheduling discipline provide differential services to each connection?
- Are there bounded measures on switch latencies and delay jitter in packet transmission at each switch node for both unicast and multicast traffic flows?
- Are the packets of a connection transmitted in the same order as when they arrive?
- Are enough buffers provided in each switch node such that any blocked packets would not be dropped?

All these issues are explained in detail in following sections. The design difficulties with switch implementations, issues with multicast traffic support, and integrated scheduling of unicast and multicast traffics are explained with respect to fundamental switch architectures.

5.3.1 Switch Architectures.

The support for connection-oriented transmission services, the real-time performance of traffic flows, either unicasting or multicasting, are mainly decided by switch architecture and service discipline of switch fabrics.

Among different switch fabrics, crossbar network is mostly preferred by switch designers since it is based on mesh network, and thus, it has no internal packet blocking inherently. Output queued (OQ) and input queued (IQ) are the two fundamental structures of crossbar switch.

5.3.1.1 Output-Queued Switch Architecture.

In OQ switches, incoming packets are multiplexed over a fast bus or any other broadcast medium. Corresponding to each output link, there is an address filter that reads the bus, selects the packets destined for that output link, and copies those packets into the corresponding output buffer. This design gives a better buffer utilization than the distributed buffer design, but a lower buffer utilization than the shared buffer design.

- Advantages of OQ architecture:
  - Absence of input contention points makes it inherently nonblocking.
  - Sophisticated QoS can be implemented by output schedulers.
  - Easy to support multicasting traffic.
- Disadvantages of OQ architecture:
  - Memory bandwidth required is N times line speed, in an N-by-N switch.
  - Current memory technology limits the size of memory elements.

5.3.1.2 Input-Queued Switch Architecture.

In IQ switches, packets arriving on each input line are placed into smoothing buffers prior to placement on the correct output lines. In any given time slot, leading packets in all the buffers are sent to their output lines. If several buffer-leading packets contend for the same output, only one of them is selected according to the contention scheme, while the rest remain in the buffers and contend again in the next time slot.

The IQ switch suffers from an unfortunate phenomenon known as head of line (HOL) blocking. The effect occurs when a packet in any given buffer is denied access to its output line, even though there are no other packets requiring the same output line, simply because the packet in the head of that buffer was blocked in contention for a totally different output. If the leading packet output line is under heavy contention, other packets in that buffer might have to wait for a relatively long period. In fact, the delay for a given packet may grow unbounded even for an offered load less than 100%.

- Advantages of IQ architecture:
  - Memory bandwidth required is same as line speed.
- Disadvantages of IQ architecture:
  - Blocking within crossbar switches and the need for an optimal matching algorithm.

- Frequency and complexity of centralized arbitration does not scale with size and interface rates.

- Arbitration determines observed bandwidth, difficult to guarantee switchwide QoS to individual flows.

- Difficult to support multicast traffic.

5.3.1.3 Switch Design Complexities.

The designers should understand switch complexities with respect to design and implementation limitations with the current technology. These complexities are mainly due to the speedup factor of a switch fabric and implementation limits on the size of memory buffers required for queuing.

A switch with a speedup of S can remove up to S packets from each input and deliver up to S packets to each output within a time slot, where a time slot is the time between packet arrivals at input ports.

In an N-by-N OQ switch, the number of packets that want to enter a given output buffer in one packet time can be as large as the number of input lines N. Accordingly, the number of input lines will be limited by the speed of the electronics used for the output buffers. Designers can avoid this limitation by limiting the number of packets that can be transferred into an output buffer to some value $K < N$. If $M > K$ *out of* N, packets are destined to the same output buffer in one packet time, then K of them are selected for transfer and the others must wait for the next available transmission slots. For avionics applications that have hard real-time communication requirements, the delay in the switch must be bounded.

An IQ *N-by-N* switch requires N^2 packet routing elements and N^2 contention elements, which usually can all be synthesized on a single very large-scale integration chip. Each of the buffers, on the other hand, normally requires a separate chip, since RAM technology normally allows the contents of only one memory location to be read or written at a time. Therefore, the switch complexity, measured as the number of chips required for implementation, grows with N.

5.3.2 Performance-Guaranteed Service Disciplines.

Service discipline is the most significant factor in guaranteeing deterministic performances of packet-based switches. The following discussion assumes that packages of each connection follow a bursty model with the parameters of average rate and burstiness, (σ, ρ). A session traffic flow is said to satisfy the (σ, ρ) model if there are at most $\sigma+\rho t$ units of traffic during any interval t. According to the definition of arrival curve, the statement that "a traffic flow satisfies (σ, ρ) model" has the same meaning as the traffic flow is constrained by service curve $\sigma+\rho t$. For example, traffic flow coming from the traffic regulator can satisfy the (σ, ρ) model.

Generalized processor sharing (GPS) is the idle rate model with absolute fairness. However, GPS cannot actually be implemented, since it is based on a pure flow model with the assumption of infinite divisibility of traffic, i.e., the transmission of a message can be pre-empted in any fine

granularity. GPS is often used as a benchmark to evaluate the characteristics of a practical scheduling discipline.

A number of work-conserving service disciplines aimed at guaranteeing deterministic communication performances have been proposed using OQ switches [15]. All of these realizable disciplines can be regarded as packet-based GPS in a wide sense [16]. Almost all of these service disciplines face a compromise between computational complexity and bound tightness. Low computational complexity is important since a switch scheduler works in real time at high speed. At the same time, real-time traffic may have relatively stringent performance requirements that require switching networks to offer tight QoS bounds. Generally, there are two kinds of work-conserving service disciplines, round-robin (RR) based and deadline based. RR-based disciplines, such as Weighted RR [17], Deficit RR [18], and Elastic RR [19], have the low computational complexity as $O(1)$, although they cannot guarantee satisfying worst-case performance bounds. Deadline-based service disciplines, such as virtual clock [20], weighted fair queuing [21], worst-case fair weighted fair queuing [22], and delay earliest due date [23], can provide relatively tighter QoS bounds desired for real-time traffic, but unfortunately, they require computational complexity as high as $O(N)$.

Another class of service disciplines is rate-controlled service (RCS) disciplines, which are primarily non-work-conserving service disciplines [24]. They are composed of a rate controller and a scheduler. The rate controller allocates bandwidth and controls traffic distortion, whereas the scheduler allocates service priorities to packets and controls the delay bounds of connections [25]. Traffic from each connection is reshaped at every node to ensure that the traffic offered to the scheduler arbitrating local packet transmissions conforms to specific characteristics. Traffic reshaping at each node results in a more predictable traffic pattern at each scheduler, which in turn facilitates the specification of delay bounds at each scheduling point. In such a setup, the end-to-end delay can be computed as the sum of the worst-case delay bounds at each intermediate node. Earliest deadline first (EDF) and first-come, first-served (FCFS) policies among others belong to this class of service disciplines.

Further, RCS decouples the allocation of the bandwidth and the delay bounds associated with a flow. For the real-time traffic that is being considered (i.e., traffic that is primarily comprised of low delay low bandwidth flows), this decoupling is crucial. For RCS disciplines, the rate controller allocates the bandwidth and controls traffic distortion while the scheduler controls the delay. Regulators also eliminate any jitter between arrivals at each intermediary node. It uniformly distributes the allocation of buffer space inside the network (among all nodes between the source and the destination). This means that, on average, each switch requires less buffer space for the connection.

Another advantage is that the regulator and scheduler mechanisms can be mixed and matched, implying that two combinations can be deployed: one for the end-systems, where the traffic to be handled is smaller, and another one for the switches, where the traffic is much larger in volume. All that is needed to be guaranteed is that the traffic pattern presented to the scheduler at each node is the same as that arriving at the first switch and that the scheduler at each switch guarantees the local delay bounds.

RCS service disciplines are more flexible and typically have a simpler implementation compared to GPS-based service disciplines. In addition, the RCS discipline that uses the non-pre-emptive version of the EDF scheduling policy can out perform GPS-based disciplines in a networked (multinode) environment, with proper choices for the traffic shaper envelope at each node.

5.3.3 Multicast Traffic Support in Switch Architectures.

For transferring multicast traffic in switching networks, there are a great number of considerations in terms of architectures and schedulers in switch design.

In general, multicasting means that a single source can send packets to multiple receivers, essentially, a subset of those that receive a broadcast packet. In a switch, this implies that an incoming packet is forwarded to more than one output port.

Multicast traffic support can be added in switches by two ways:

- Multicasting can be implemented by transferring the same packet from the sender to multiple receivers through multiple transmissions as unicast does.

- Special switch architectures supporting multicast traffic can also be designed.

5.3.3.1 Multicasting in OQ Architectures.

The output buffer design of OQ switches is best suited for multicast traffic. Indeed, packets with multiple destination ports can be handled with virtually no additional complexity.

To support multicasting in OQ switches, in addition to the *N* bus interfaces, the input buses are attached to, at most, *M* multicast modules specially designed to handle multicast packets. Each multicast module has *N* inputs and one output. The inputs are the signals from the input interface modules. The output drives one of *M* bus wires as part of the arrangement for broadcasting to all the *N* bus interfaces.

These multicast modules can be implemented by two ways: (1) using fast packet filter and (2) adding copy network for packet duplication. When using a fast packet filter, once a multicast packet is a winner from the knockout concentrator, the packet is broadcasted directly to all the bus interfaces without any modification. It is the packet filter's responsibility to determine whether an arriving packet is destined for its output, and this has to be done in a few bit intervals so that no excessive delays are introduced. When using the approach of packet duplication, the incoming packets are selected through some filtering module such that only multicast packets are retrieved and forwarded to the packet duplicator. The duplicated packets are sent along the broadcast bus to the required bus interfaces.

Both the fast packet filter and the packet duplicator approaches conform well to the original knockout switch architecture. Each guarantees the first-in, first-out (FIFO) packet sequence for the multicasting packets. As far as delay is concerned, the fast packet filter technique is clearly better, particularly so in cases where a multicast packet has a great number of destinations. Therefore, the fast packet filter is definitely favored in applications where the multicast traffic is

heavy and involves a large number of simultaneous destinations for each packet. However, the packet duplicator method does not require distributed circuit tables and is somewhat easier to manage in terms of circuit updates. Thus, it seems to be more appropriate in situations where heavy multicast traffic is not demanded.

5.3.3.2 Multicasting in IQ Architectures.

The IQ switch design does not cope well with the multicast traffic. If a head packet is of this kind, it has to contend simultaneously for all the outputs it is intended for. HOL blocking can be aggravated many times if the contention schemes are independent. Alternatively, the RR contention policy can be implemented in such a way that the rotating pointer is the same for all outputs at any given time. Such an implementation guarantees that a packet does not stay in the queue for a longer time just because it is a multicast packet, but it leads to an even larger delay for the normal unicast packets.

5.3.3.3 Multicast Switches.

Implementing multicasting using the unicast approach results in too much traffic in the system. To avoid excessive traffic, the switch architecture can be designed beforehand to address the multicast traffic. These switches are called as multicast switches.

Multicast switches, which are suitable for conveying multicast traffic, include a packet-replicating function. A switch with the capability of packet replication is called a cascaded multicast switch and it is composed of two stages: copy network and unicast network. The copy network replicates the packets destined for multiple output ports. The replicated packets then enter the unicast network where all packets are forwarded to output ports in a unicast format.

As far as scheduling disciplines of multicast switches are concerned, there are two basic strategies that can be followed: (1) nonfanout splitting and (2) fanout splitting. Fanout is defined as the number of different destinations to reach by a multicast packet. The fact that a multicast packet has multiple destinations implies that some scheduling disciplines may elect to transfer the multicast packet to all destinations in a single transfer operation, while others may elect to transfer the packet in several transfer operations, reaching non-overlapping and exhaustive subsets of destinations. In the case of partial service, the residue is defined as the set of fanout destinations that have not yet been reached after a multicast packet is transferred towards output ports. These two options are defined as:

- Nonfanout Splitting: Any multicast packet is transferred through the switching fabric once, only when all fanout destinations can be reached in the same time slot. If any of the fanout destinations cannot be reached because the multicast packet loses contention for an output port, the packet cannot be transferred, and it will contend again in a future time slot. As a consequence, this discipline favors packets with small fanout.

- Fanout Splitting: Multicast packets may be delivered to output ports over a number of time slots. Only those fanout destinations that could not be reached in previous time slots are considered in the next time slot. Although fanout splitting is better than nonfanout

splitting in terms of throughput, nonfanout splitting may be better than fanout splitting in terms of jitter control.

5.4 DATA INTEGRITY AND RELIABILITY.

Data integrity refers to the degree of correctness of the data transferred between entities residing on the network. Data integrity, sometimes identified with reliability, can be considered at two levels, one at the system level and the other at the message level.

5.4.1 System-Level Reliability.

At the system level, reliability refers to the ability of the system/network to withstand node and link failures. In this sense, one can think of it as the degree of fault tolerance of the network.

Any implementation of a deterministic network should provide some degree of fault tolerance and fault effect isolation. Fault effect isolation is to filter the fault effects within a network that could compromise the deterministic communication between two fault-free, end-systems. The method of fault effect isolation should be tailored to meet the safety requirements of a particular network and the set of applications. Fault isolation in Ethernet-based communication framework can be achieved with call admission control, bandwidth reservation, and traffic regulation policies.

Physical redundancy can be used to increase the degree of fault tolerance of the system. The presence of a redundant network ensures that communication between end-systems is protected against the failure of any network component such as a link or a switch.

In designing a fault-tolerant system, one realizes that 100% fault tolerance cannot be achieved. Moreover, the closer one wishes to get to 100%, the more costly the system will be, either in hardware or software redundancy. To design a practical system, one should consider the degree of replication needed. This will be obtained from a statistical analysis for probable acceptable behavior. Factors that enter into this analysis are the average worst-case performance in a system without faults and the average worst-case performance in a system with faults.

Some design issues to be addressed:

- Does the network ensure losslessness?
  - If the network is lossy, what is the loss ratio? (number of bytes/Mb)
  - How is the loss of a packet notified to the application? (sender/receiver)
  - Is there an error recovery mechanism for lost packets? If so, what is the overhead generated by it?
  - How is it ensured that packets are not lost due to collisions on the databus?

- How is it ensured that packets are not lost due to the buffer overflows in the end-system communication architecture?

- How is burstiness of the traffic (generated by an application) handled?

- How is it ensured that packets are not lost in the presence of hardware (end-nodes/intermediate nodes/links) failure?

- How do you ensure that the intended recipient receives the data? What is the addressing scheme used?

- Does the network ensure fault isolation?

  - How is it ensured that an erroneous connection does not disrupt the operation of the error-free connections in the system?

  - How is the starvation of low bandwidth connections (due to a high bandwidth connection) prevented?

  - How is overallocation of bandwidth leading to system overload and, hence, loss of packets prevented?

  - How is an application's compliance to its negotiated QoS parameters monitored and controlled?

  - Does the system contain a diagnostic mechanism to detect faults and to isolate them?

5.4.2 Message-Level Reliability.

Message-level reliability guarantees that messages are delivered without any error, or any introduced error is detected with high reliability when received.

Message-level reliability can be achieved through information redundancy by using various error detection and control mechanisms such as cyclic redundancy codes or forward error correction mechanisms. For example, forward error correction mechanism is used to anticipate errors and provide information redundancy, allowing the receivers to reconstruct the information without asking the sender to retransmit. It improves data reliability by introducing a known structure into a data sequence prior to transmission or storage. This structure enables a receiving system to detect and possibly correct errors caused by the corruption.

Alternatively, time redundancy, where the same operation is performed multiple times, can be deployed in the system. For message-level reliability, time-redundant mechanisms, such as message retransmission, acknowledgement, and sequence control policies, can be implemented.

Design-level issues to be addressed for message-level reliability are described in section 5.5.

5.5 NETWORK STACK PROCESSING.

The network protocol stack is primarily used for handling data transmission/reception. The protocols (above Ethernet layer) used in a network stack can have a great impact on the determinism of data delivery. All the characteristics of these protocols should be analyzed to ensure that it would not harm the determinism of the system. This section considers three typical subjects in network stack processing and the related design issues to be addressed.[1] Additionally, recommendations for network stack implementation are provided.

5.5.1 Use of Connection-Oriented or Connection-Less Protocols.

Protocols within a network stack can be either connection-oriented or connection-less in nature. In connection-oriented protocols, communicating nodes maintain state information about the dialogue they are engaged in. This connection state information supports error, sequence, and flow control between the corresponding entities, as described below.

- Error control refers to a combination of error detection, error correction, and message acknowledgment sufficient to compensate for any unreliability inherent within the communication channel. The packet header should support some additional information to detect errors that occurred during the transmission of the packet. Cyclic redundancy check, packet checksum, etc., are the most general-purpose mechanisms being deployed for error detection. Extra information can be put forth in the packet header to recover the corrupted data. For instance, a forward error correction mechanism can be used to anticipate errors and provide information redundancy, allowing the receivers to reconstruct the information without asking the sender to retransmit. Also, the message acknowledgement policies guarantee the successful packet transmission over the channel, otherwise lost packets can be retransmitted. Also, the separate error control protocols (e.g., Internet control message protocol) can also be implemented within the system to notify the critical errors to other nodes.

  The error detection, error correction, message acknowledgement, and retransmission mechanisms increases the reliability of the system, besides the redundancy provided by the physical layer. However, because of the extra information added in the header, these mechanisms results in extra transmission overhead per packet, thus reducing the system throughput. The retransmission policies cause extra transmission overheads and increase unnecessary delays. The deployment of retransmission policies results in packet duplication at the current layer and reduces bandwidth utilization of the system.

- Sequence control refers to the ability of each node to reconstruct a received series of messages in the proper order. Thus, the inclusion of sequence numbers in packet headers guarantees the in-order delivery of packets in the system. Also, most of the message retransmission mechanisms are based on these sequence numbers, where the node resends all the packets following the lost packet just to guarantee the in-order delivery of all packets. However, the addition of sequence numbers and corresponding impact on the

[1] Not “all” issues are included in this section, since there are typically project-specific issues that arise. This section merely provides the typical/general issues that should be considered.

retransmission policies induces extra transmission overhead per packet and nondeterminism in the system. The retransmission policies should come up with the buffer requirements and should put the upper bounds on the number of packets being retransmitted in the current window of acknowledgement.

- Flow control refers to the ability of both source and destination nodes in a dialogue to avoid overrunning their peer with too many messages. The system should guarantee that the flow specification is not violated and violation of flow specification by one flow should not affect other flows. The issues dealing with the flow control mechanism are discussed in section 5.2.

Connection-oriented protocols also result in extra overhead related with session management. It includes connection establishment and connection release overheads.

Transmission by using connection-less protocol does not need to setup any end-to-end connection with the receiver node. Connection-less communication is usually achieved by transmitting information in one direction, from source to destination without checking to see if the destination is still there, or if it is prepared to receive the information. Due to lack of connection establishment, it requires a less overhead than connection-oriented transmission. Therefore, connection-less transmission is best when data is generated in intermittent short bursts.

The designer should prioritize the communication requirements, such as reliability, data accuracy, bandwidth utilization, and system throughput, in order to decide whether connection-oriented or connection-less protocols are the best choice for the system.

If reliability and data accuracy is paramount, then connection-oriented protocol is the best option. However, if the delay bounds and jitter requirements (also bandwidth utilization and system throughput) are of primary concerns, then connection-less protocols can be considered as the better choice. If the designer decides to deploy the connection-less protocols, then also depending on the application requirements, it should be specified at what extent the error control, sequence control, and flow control mechanisms are being supported over the connection-less protocols.

The following issues should be addressed, at minimum:

- How reliable should the connection be?
- Is the in-order delivery of packets required on the receiver side?
- Should it have flow control?
- Should it acknowledge the messages it receives?
- Is the retransmission of packets required?

- If retransmission is to be implemented, how does it affect system delay bounds and buffer requirements?

- Should it be able to handle duplicate data packets?

- Are the external error control protocols being deployed in the system? Because of this, are the system performance metrics within bounds?

- What kind of service can the application live with?

- What level of performance is required?

5.5.2 Address Resolution Mechanisms.

Address resolution mechanisms are used to translate the logical network address, which are specific to one protocol, to a corresponding logical/physical address. This mechanism is generally used in multilayer network protocol stacks. For instance, address resolution protocol (ARP) is used in the IP/Ethernet stack to resolve IP address into the corresponding media access control (MAC) address.

The use of address resolution mechanisms helps to resolve the network address dynamically. But this dynamic nature of the protocol and the underlying repetitive broadcasting mechanism causes variable times for address look-up and nondeterministic operations in the system.

To achieve complete determinism for address resolution, two options are recommended:

- The system can be configured statically. The designer must have the knowledge of all the nodes within a network. Each node should have the corresponding logical and physical address mapping tables stored during the system initialization. Thus, at run time, the ARP implementations can guarantee the worst-case address resolution time and constant look-up times.

- The ARP should address all inherent nondeterministic properties such as variable address resolution time, use of broadcasting mechanism for address resolution requests, nonguaranteed operations, and other performance metrics bounds such as address resolution latency and address resolution jitter.

Design issues to be addressed are:

- Is the address resolution mechanism really required? (Is it multilayered network protocol stack architecture?)

- Is the worst-case address resolution time guaranteed?

- If the system is not configured statically and address resolution is dynamic in nature, then does it guarantee the address resolution jitter to be in the specified bounds?

- Does the address resolution guarantee the successful operation?

- If address resolution is carried out over the broadcasting channel, then how does it affect bandwidth utilization and system throughput?

5.5.3 Networking Address Modes.

Most of the application traffic generated can be characterized as unicast or multicast. Unicast traffic, also known as point-to-point, is common in most QoS-guaranteed applications. However, many applications produce multicast traffic, requiring that the same piece of data from a sender be transmitted to multiple receivers.

Network stack should provide the support for multicast and corresponding multicast addressing, depending on application requirements and system performance metrics.

Design issues to be addressed, at minimum, are:

- Does the network stack require the support of multicast traffic?

- Does the underlying hardware and Ethernet layer (driver) support multicast traffic?

- If yes, does the network stack exploit the multicast support from underlying layers?

- Where is the multicasting addressing supported? (Ethernet layer MAC multicast addresses or network layer, e.g., IP multicast addresses)

- If the multicast addressing mode is supported at higher layers (along with the Ethernet layer), how does it balance the system performance metrics (bandwidth utilization versus multicasting setup/transmission overhead)?

- Multicasting requires the generation of multicast trees. If the multicast addressing mode is supported at higher layers above Ethernet layer (e.g., IP), then which layer generates the multicasting trees?

- How does the protocol guarantee to provide optimal multicast trees?

- Are the multicast trees generated statically or dynamically?

- If multicast trees are generated dynamically, how does it guarantee the performance metrics bounds (latency, delay-jitter, buffer requirements, etc.)?

5.6 NONDETERMINISM IN HARDWARE COMPONENTS WITHIN END-SYSTEM.

The network controller and underlying hardware interfacing bus are under the control of a host processor and can introduce various delay factors in the communication system. These delay factors are due to interrupt latency, the latency of bus access and arbitration, memory read/write operations, and buffer management policies. Also, most of the current Ethernet controllers use

direct memory access (DMA) capabilities that reduces the central processing unit (CPU) involvement in transferring large amounts of data between the main memory and I/O devices.

The hardware interfacing buses like Peripheral Component Interconnect (PCI), Industry Standard Architecture, etc., are initially not developed to accommodate time-critical applications and do not guarantee constant data transfer rate at all times. Industrial testing shows the bus performance is very complex and, consequently, hard to predict. Thus, in a real-time system, where communication tasks are bound by these buses and DMA capabilities, the hardware interfacing bus can be a bottleneck of the Ethernet Network Interface Card (NIC) in terms of throughput and latency. For example, PCI-DMA Ethernet controllers transmit and receive data frames by making use of interrupt mechanism, burst reading, and writing to PCI devices (DMA operations). These result in some nondeterministic features to real-time communications.

To ensure the deterministic communication performance of Ethernet controllers, factors contributing to its nondeterminism should be carefully analyzed. A PCI bus will be used as an example to illustrate the design issues to be addressed. In general, for these design issues, a thorough investigation must be done such that the worst-case delays can be identified and their impacts are analyzed. Databus developers should also identify and address specific issues related to other buses by following corresponding bus specifications.

The design issues to be addressed in a PCI-based Ethernet controller are, at a minimum:

- Can the worst-case bound of interrupt chaining latency be identified?
- Can the worst-case bound of interrupt latency caused by context switch be identified?
- Can the PCI bus arbitration and access latencies be analyzed?
- What is the worst-case delay on DMA transfers via virtual memory and noncontiguous buffers?
- What is the worst-case delay on interrupt service routines due to non-pre-emptive DMA cycles?
- For the management scheme of the transmit/receiver buffers in an Ethernet controller, is the worst-case message processing delay by the controller bounded?

A detailed consideration of these issues for Ethernet controllers on a PCI bus is presented in appendix A.

5.7 SYSTEM CONFIGURATION.

It is very important for a system to provide a mechanism whereby it can be configured, and the way a system is configured can make a lot of difference to its performance and efficiency.

To achieve the determinism of an aviation databus, the communication system should be analyzed statically and all configuration parameters should be identified.

Configuration parameters for a databus system may include facilities to specify:

- Type of applications that can communicate through the databus
- Number of such applications that can communicate simultaneously
- Amount of bandwidth reserved for control data
- Flow specification for each application

In a communication subsystem, the configuration parameters of each component should be well identified in advance. If the component is multilayered, e.g., a network stack, then the configuration parameters of each layer should be specified at design time.

These parameters can be

- Number of nodes in the whole system
- IP address configuration of the network interfaces
- ARP address mappings
- Routing table configurations
- Configuration of number of hops information for each target in the network
- Network buffer memory in switches, Ethernet driver, etc.
- Maximum number of connections allowed in the system
- Network interface initialization parameters (hardware address/broadcast address/ broadcasting style)

The system can also be reconfigured dynamically. Dynamic reconfiguration of the system changes the system configuration parameters to achieve optimal performance, cope with component failures, and improve the scalability at run time.

When reconfiguring the system dynamically, the databus developers should address the following issues, at a minimum:

- Controllability of the modifications/additions of nodes and/or applications on the bus is necessary.
- The configuration parameters that can be changed at run time should be specified.
- The configuration parameters for those that cannot be changed at run time should be specified and the system should ensure that these parameters will not get changed indirectly during reconfiguration.
- The effect of change of a particular parameter in the system should be documented.

- The system should ensure that the dynamic reconfiguration of some connection flows will not affect other connection flows in any respect, like changes in their service rates, delay jitter, latencies, etc.

- The dynamic reconfiguration of the session parameters may induce nondeterminism in the end-system. Hence, it is required that the underlying communication subsystem must provide the guarantee of deterministic behavior of the system, if it is supporting session reconfiguration.

- Definite interfaces for reconfiguration should be provided.

- Requests made by the application for configuration purposes (e.g., route entry in the routing table) should have bounded service times.

- Reconfiguration services should be protected such that it will be available only to special purpose applications such as network administration functions.

6. SUMMARY.

This Handbook provides guidelines to network designers and developers to develop an Ethernet-based aviation databus. Its objective was to facilitate the process by which an Ethernet-based databus may be qualified as part of an overall aircraft certification. It focused on identifying aspects of the product that detract from the factors impacting qualification.

Based on the certification authorities' CAST Position Paper 16, the Handbook identifies the certification issues related to an Ethernet-based avionics databus. The issues are divided into two groups: the first group consists of the ones that are generic in nature and must be evaluated regardless of the specific databus technology selected. The other group is made of the remaining criteria that are application- and/or implementation-specific and can only be evaluated in the context of a specific databus technology and network topology. This Handbook elaborates on and details the evaluation criteria for general aviation databus for an Ethernet-based avionics databus. The designers and developers are advised to understand these criteria and should follow these guidelines in early phases of the development cycle so that a certifiable Ethernet databus solution can be designed.

For avionics databus operations, the Handbook defines determinism for a communication system as the ability of the network to guarantee delivery of messages to the receiving nodes within a specified period of time. This definition can be extended to the cases of bit errors in transmission and hardware/software failures in terms of message drop probability and bounded error detection/recovery latency. However, due to the variations in message arrival processes of flow connections, message lengths, and the behavior of software and hardware components, there are nondeterministic factors that must be resolved before deploying Ethernet in aircraft networks. The certifiability can then be accomplished when the required characteristics can be demonstrated by proof and extensive testing.

The Handbook discusses design issues by illustrating how an Ethernet-based databus can provide guaranteed services. At end-systems, both network stack software and Ethernet adapter should be investigated. The design considerations include flow (traffic) management, error/sequence control, and the potential interference of shared hardware mechanisms. At the network level, the protocol to be deployed must avoid collision, and the switching mechanisms must guarantee bounded delay through provable message scheduling algorithms. From a system's point of view, the Handbook brings up the design concerns for configuration, data integrity, and reliability. Through the discussion of these issues, the Handbook provides an illustration on how an Ethernet-based databus can provide guaranteed services and meet the deterministic requirement for aviation applications.

7. REFERENCES.

1. FAA CAST-16 Position Paper, "Databus Evaluation Criteria," February 2003 http://www.faa.gov/aircraft/air_cert/design_approvals/air-software (Refer to CAST page).

2. Lee, Y.H., Rochlin, E., and Scandura, P.A., "Safety and Certification Approaches for Ethernet-Based Aviation Databuses," FAA report DOT/FAA/AR-05/52, 2005.

3. Galotti, Vincent, P., "The Future Air Navigation System (FANS)," Avebury, Ashgate, Publishing Company, 1997, pp. 3, 8, and 62.

4. "Backplane Databus," ARINC Specification 659, Aeronautical Radio Inc., Annapolis, MD, December 1993.

5. "Ethernet Physical and Data Link Layer Specification," ARINC Specification 664 – Part 2, Aeronautical Radio Inc., Annapolis, MD, January 2003.

6. ARINC ADN 664: Aircraft Data Network, Part 7–Avionics Full Duplex Switched Ethernet (AFDX) Network, July 2005.

7. SAE/ARP4761, "Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment," December 1996.

8. RTCA/DO-254, "Design Assurance Guidance for Airborne Electronic Hardware," April 2000.

9. RTCA/DO-178B, "Software Considerations in Airborne Systems and Equipment Certification," December 1992.

10. FAA Order 8110.4B "Type Certification," April 24 2000. http://www.airweb.faa.gov/Regulatory_and_Guidance_Library/rgOrders.nsf/EA40C6F8F194E49086256F8EC076ACB2?OpenDocument

11. FAA Aircraft Certification Design Approvals: Original Design Approval Process "The FAA and Industry Guide to Product Certification," Second Edition, September 2004. http://www.faa.gov/aircraft/air_cert/design_approvals/media/CPI_guide_II.pdf

12. "Avionics Application Software Standard Interface," ARINC Specification 653, Aeronautical Radio Inc., Annapolis, MD, January 1997.

13. RFC-1363 "A Proposed Flow Specification," Internet Engineering Task Force, September 1992.

14. RFC-2215 "General Characterization Parameters for Integrated Service Network Elements," Internet Engineering Task Force, September 1997.

15. Zhang, Hui, "Service Disciplines for Guaranteed Performance Service in Packet-Switching Networks," *Proceeding of the IEEE*, Vol. 83, No. 10, October 1995.

16. Parekh, Abhay K. and Gallager, Robbert G., "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: the Single-Node Case," *IEEE/ACM Transactions on Networking*, Vol. 1, No. 3, June 1993.

17. Katavenis, M., et al., "Weighted Round-Robin Cell Multiplexing in a General-Purpose ATM Switch Chip," *IEEE Journal on Selected Areas in Communication*, Vol. 9, No. 8, 1991.

18. Shreedhar, M. and Varghese, George, "Efficient Fair Queuing Using Deficit Round-Robin," *IEEE/ACM Transactions on Networking*, Vol. 4, No. 3, June 1996.

19. Kanhere, Salil S., Sethu, Harish, and Parekh, Alpa B., "Fair and Efficient Packet Scheduling Using Elastic Round Robin," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 13, No. 3, March 2002.

20. Zhang, Lixia, "VirtualClock: A New Traffic Control Algorithm for Packet Switching Networks," *Proc. ACM SIGCOMM'90*, Philadelphia, PA, September 1990, pp. 19-29.

21. Demeres, S. Keshav and Shenker, S., "Analysis and Simulation of a Fair Queueing Algorithm," *J. Internetworking Res. and Experience*, October 1990, pp. 3-26.

22. Bennet, J.C.R. and Zhang, H., "WF2Q: Worst-Case Fair Weighted Fair Queuing," *Proceeding of IEEE INFOCOM'96*, CA, March 1996, pp. 120-128.

23. Ferrari, D. and Verma, D., "A Scheme for Real-Time Channel-Establishment in Wide Area Networks," *IEEE Journal on Selected Areas in Communication*, April 1990.

24. Zhang, H. and Ferrari, D., "Rate-Controlled Service Disciplines," *Jour. High Speed Networks*, pp. 482-501, 1994.

25. Geogradis, L., Guerin, R., Peris, V., and Sivarajan, K., "Efficient Network QoS Provisioning Based on per Node Traffic Shaping," *IEEE/ACM Trans. Networking*, August 1996, pp. 482-501.

8. GLOSSARY OF TERMS.

This section provides key definitions and acronyms used throughout this Handbook.

ADS-B: Automatic Dependent Surveillance-Broadcast. Equipped aircraft and vehicles automatically broadcast important information, via data link, such as latitude and longitude, velocity, altitude, heading, and identification, as determined by the avionics on board and the Global Navigation Satellite System.

ARP: Address Resolution Protocol. The term address resolution refers to the process of finding an address of a computer in a network. The address is resolved using a protocol in which a piece of information is sent by a client process executing on the local computer to a server process executing on a remote computer. The information received by the server allows the server to uniquely identify the network system for which the address was required and, therefore, to provide the required address. The address resolution procedure is completed when the client receives a response from the server containing the required address.

COTS: Commercial Off-The-Shelf. Refers to a set of products that are commercially available for integration into larger systems as its elements.

CPU: Central Processing Unit. The central processing unit, or CPU, is a dated term synonymous with processor, microprocessor, or chip. To anthropomorphize computers for a moment, the processor is the machine's brain. It takes in instructions, processes them, and pumps out responses.

CSMA/CD: Carrier Sense Multiple Access/Collision Detection. CSMA/CD is the protocol for carrier transmission access in Ethernet networks. On Ethernet, any device can try to send a frame at any time. Each device senses whether the line is idle and, therefore, available to be used. If it is, the device begins to transmit its first frame. If another device has tried to send at the same time, a collision is said to occur and the frames are discarded. Each device then waits a random amount of time and retries until it is successful in getting its transmission sent.

DMA: Direct Memory Access. Refers to the term used for a device/controller that transfers data to and from memory without complete intervention of the central processing unit.

EFIS: Electronic Flight Instrument System. Refers to one functional-area/system in the FMS. A full EFIS combines traffic and resolution advisory information from different sources within the aircraft and provides the pilot with a unified display. This greatly simplifies the instrument scan and improves positional awareness.

FANS: Future Air Navigation System. FANS will provide the aviation industry with solutions to enhance safety and alleviate congestion, eliminating economic losses generated through delays. FANS is a concept that will meet the need of the civil aviation community over the next century, and it is also a systems concept to achieve this need, which is called the Communications, Navigation, Surveillance/Air Traffic Management (CNS/ATM) systems concept. The FANS CNS/ATM system concept involves a complex and interrelated set of technologies that is dependent mainly on satellites and was endorsed by the International Civil

Aviation Organization in 1991. From the communications perspective, FANS will include communication protocols based on open systems interconnection (OSI) procedures and allowing the interconnection of airborne, ground and air/ground networks, including local area networks (LANs), wide area networks (WANs) and radio links allowing a seamless, transparent, worldwide aeronautical digital data communications network. One of the possibilities for constituting these LANs and WANs will be Ethernet.

FMCS: Flight Management Computer Systems. Refers to a system that controls/monitors most of the flight-related parameters.

IEEE: Institute of Electrical and Electronic Engineers. An international standards development organization.

IP: Internet Protocol. Refers to the protocol used in the network layer.

ISA: Industry Standard Architecture. The bus used in standard IBM-compatible PCs to provide power to add-in boards and communication between the add-in boards and the motherboard (into which the boards plug).

ISR: Interrupt Service Routine. Routine that handles the interrupt.

LAN: Local Area Network. LAN is a system that connects computers and other devices within the same physical proximity to form a network, usually with a wiring-based cabling scheme. LANs connect personal computers and electronic office equipment, enabling users to communicate, share resources such as data storage and printers, and access remote hosts or other networks.

MAC: Media Access Control. Refers to the layer between the data link and network layer in the standard OSI architecture.

NIC: Network Interface Card. A name for the LAN adapter (printed circuit board), installed in a PC, that enables it to communicate over a LAN. The term is used more often by IBM customers and Token Ring people.

OSI: Open System Interconnection. Suite of protocols and standards sponsored by the International Standard Organization (ISO) for data communications between otherwise incompatible computer systems.

PCI: Peripheral Component Interconnect. Intel's local bus standard. Introduced in 1992 (the first version) and 1993 (Release 2.0). Supports up to 16 physical slots—an addressing limitation which normally will not be reached because of the electrical limitation of 10 loads (which will typically amount to three or four plug-in PCI cards) residing on each PCI bus. PCs can have two or more PCI buses, so there can be six or more PCI cards per PC.

PHY: Physical Layer. Refers to a layer in OSI.

QoS: Quality of Service. The network requirements (latency, maximum packet loss, etc.) to support a specific application.

RFC: Request For Comment. The RFC document series is a set of technical and organizational notes about the Internet (originally the ARPANET), which began in 1969. Memos in the RFC series discuss many aspects of computer networking, including protocols, procedures, programs, and concepts. The official specification documents of the Internet Protocol suite that are defined by the Internet Engineering Task Force (IETF) and the Internet Engineering Steering Group (IESG) are recorded and published as standards track RFCs. An RFC can have several classes, including Informational, Experimental, Proposed standard, Standard. Once published, RFCs are never modified or changed, only superceded.

TCP: Transmission Control Protocol. Refers to the transport layer protocol in OSI.

UDP: User Data-gram Protocol. Refers to the transport layer protocol in OSI.

VLSI: Very Large-Scale Integration. Integrated circuits with 100,000 to 1 million transistors per chip.

APPENDIX A—NONDETERMINISTIC OPERATIONS OF ETHERNET NETWORK INTERFACE CARD ON A PCI BUS

A.1 INTERRUPT CHAINING LATENCY.

According to the Peripheral Component Internet (PCI) specification [A-1], "the system vendor (computer vendor) is free to combine the various interrupt signals from the PCI connector in anyway to connect them to the interrupt controller." Hence, a device driver may have much freedom to configure the device's interrupt routing to the host machine. This may inadvertently degrade the entire system performance in light of delay and predictability, when multiple devices have interrupts associated with a same interrupt service routine (ISR) in a host machine. In other words, an interrupt chain is formed.

To illustrate the performance deterioration resulting from interrupt routing, consider a typical Industry Standard Architecture (PCI/ISA) bus design in which similar interrupt signal pins from various PCI connectors are connected (e.g., all the INTA#[*] pins are connected and all INTB# pins are connected) and directed to a programmable interrupt router. The router then routes each interrupt in the group to the corresponding ISR, which results in an interrupt chain. The flaw with interrupt chaining is an increase in interrupt latency, if a specific interrupt service routine does not stay at the head of the chain. The interrupt latency (the time elapsed from the time a device activates an interrupt to the time the corresponding ISR runs) varies, depending on where the ISR for the device is located in the chain. In the worst case, when all the interrupts in the chain are invoked simultaneously, the latency of the last interrupt would be much longer than when it is invoked by itself. It is possible that the interrupt chain can be updated at run time. In other words, some interrupts in the chain may be disabled, some may be added to the chain, and some already in the chain, but disabled, may be enabled again. All of these can happen at system run time.

A.2 INTERRUPT LATENCY FROM CONTEXT SWITCHING.

In modern operating systems, any interrupt response involves context switching between kernel mode and user mode. Depending on process scheduling and the interrupt management mechanism, there are some nondeterministic factors in context switching in terms of latency. For instance in the Linux system, the work that the ISR should do is assigned to what is called a bottom half, which is scheduled to run at a later time. The duration from the time an interrupt is triggered to the time when the bottom half is executed could be quite large, since some urgent work might have to be completed before the bottom half is run and is, thus, nondeterministic.

A.3 PERIPHERAL COMPONENT INTERNET BUS ARBITRATION AND ACCESS LATENCY.

Arbitration latency is defined as the number of bus clock cycles it takes between a master's request of the bus and when the arbiter grants the bus to that master. This period is a function of the arbitration algorithm, the master's priority and whether any other masters are requesting

[*] The notation INTA# represents a signal pin named INTA which is active when low (i.e., negative logic).

access to the bus. Thus, the arbitration latency is implementation dependent and, thus, nondeterministic.

All bus operations are serialized such that read and write, for example, cannot be executed in parallel. On the other hand, it is expected that the Ethernet controller will run in full-duplex mode for real-time communications, using separate queues for transmit and receive. In spite of the full-duplex operation of the controller, the PCI bus by which the host transmits and receives frames can only be possessed by one master device at a time. With multiple devices or masters in a bus, an access delay is expected when the bus utilization is high.

A.4 VIRTUAL MEMORY AND NONCONTIGUOUS BUFFERS.

Direct memory access (DMA) devices reference only a block of contiguous physical address. However, operating system and application programs reference virtual memory. For a DMA device to run with the virtual memory manager of an operating system, the driver software for the device must assume the responsibility of supervising the data location and managing the data transfer between physical and virtual memory. Before data can be transferred, the virtual memory manager must be instructed to map the data buffer from virtual memory to physical memory (page in). Sufficient physical memory must be available to contain the entire buffer. In addition, physical memory must be locked so that it is not moved to another memory location (for example, performing a page out, in order to meet the memory needs of another operation) while the DMA is in progress.

A DMA operation command is issued by the application program, which references the buffer in virtual memory. A DMA device (the operation executor) references contiguous physical address. But a contiguous buffer in virtual memory cannot guarantee contiguity in the physical memory. It is possible that a contiguous buffer in virtual memory consists of multiple distinct (noncontiguous) physical segments. In that case, it will take more than one DMA transaction to transmit the data in this virtual memory buffer. The time to transmit two virtual buffers of data of the same length may vary greatly. The driver has to know exactly the mapping from virtual memory buffer to the segments of contiguous physical memory. The data in a fixed buffer in virtual memory could be transferred through DMA operations in a large range of time intervals at different times due to paging on demand and physical memory fragmentation in some operating systems.

A.5 LATENCY FROM NON-PRE-EMPTIVE DMA MECHANISM.

When one DMA transaction is executing, another DMA request of an even higher priority than the executing one cannot be granted until the current transaction terminates because the bus arbiter normally does not use the pre-emption mechanism to assign the bus to master devices. DMA cycles on a PCI bus are proportional to the sizes of the data to be transferred. The larger the block of data to be transferred, the more delay that could be introduced to a latter DMA request when the former one is executing. The scenario, for example, to be considered in an Ethernet Network Interface Card, is how to coordinate the DMA priorities of transmit and receive. Obviously, transmit and receive DMA operations influence transaction latency to each other since they share the PCI address and databuses.

A.6 TRANSMIT/RECEIVE BUFFER MANAGEMENT.

Data buffers are needed to balance the data transfer rates of the microprocessor and the memory and peripheral devices. The larger the buffer, the greater are the chances of data being transmitted reliably (i.e., without packet drops due to buffer overruns).

Ethernet controllers support priority-based multiple transmit and multiple receive queues. During a receive operation, the driver should inspect the priority of the coming frame and put it to the appropriate queue. During a transmit operation, the controller would take a frame in the higher-priority queue to send out and this characteristic benefits the real-time applications. But this characteristic makes the driver somewhat more complicated than the simple first-come, first-served (FCFS) case. It is also not easy to evaluate quality of service parameters (such as jitter) in FCFS, since the influence of frame priority should be considered. The controller first-in, first-out (FIFO) buffer delay of a frame is determined by the backlog (the number of frames and frame size) in the FIFO when the frame arrives at the FIFO. With a FIFO queuing mode, it is not difficult to analyze FIFO performance. For multiple priority FIFO queues, the analysis would be complex, considering priority influence.

A.7 REFERENCE.

A-1. Intel, PCI Specification V2.1.