Academic Journal of Science and Technology
ISSN: 2771-3032 | Vol. 15, No. 3, 2025

Research and Analysis on AFDX Network Configuration Based on OMNET++ Simulation

Lei Kang, Ruixin Du

School of Xi'an Shiyou University, Xi'an 710065, China

Abstract: With the continuous improvement of data transmission performance requirements in avionics systems, traditional communication buses such as ARINC429 can no longer meet the needs of modern avionics systems for high bandwidth and real-time performance. Avionics Full-Duplex Switched Ethernet (AFDX), as a network technology in Integrated Modular Avionics (IMA), has gradually become the key to solving such requirements. This paper constructs an AFDX network simulation framework based on the OMNeT++ platform, carries out modular improvements and extensions to the framework, and optimizes key functional modules of end systems and switches, including frame parsing, scheduling mechanisms, buffer management, and performance monitoring. By adjusting network parameters such as the number of end systems, bandwidth allocation intervals, and queue lengths, the impact of different configurations on network performance is systematically analyzed. The simulation results show that reasonable parameter configuration can effectively improve the end-to-end delay and bandwidth utilization of the network, providing theoretical support and technical paths for the design and optimization of avionics networks.

Keywords: AFDX; OMNeT++; Avionics; Network Modeling; Performance Optimization.

1. Introduction

With the continuous development of avionics systems, flight missions have become increasingly complex. Avionics systems undertake critical control and communication tasks, whose performance directly affects flight safety and mission efficiency. To enhance system reconfigurability, functional integration, and resource utilization, civil and military aircraft universally adopt the IMA architecture. This architecture significantly improves system maintainability and upgrade capabilities through centralized resource management and modular design, while imposing higher real-time and deterministic requirements on communication networks. To meet these needs, ARINC664 Part 7 proposed the AFDX standard[1]. Based on full-duplex switched Ethernet, AFDX retains the advantages of Ethernet's high throughput and open standards, while introducing mechanisms such as Virtual Links (VL), Bandwidth Allocation Gaps (BAG), and Maximum Frame Sizes (Smax). These mechanisms achieve physical isolation and time constraints for communication flows, ensuring reliable transmission of critical mission data in avionics systems.

Although AFDX has been applied in multiple new-generation airliners (e.g., A380, B787), designing a reasonable network configuration for specific system task requirements remains an urgent issue to be addressed[2]. Network architecture and parameter configurations directly impact system load balancing and transmission delay. Traditional empirical methods struggle to quantify the impact of parameter configurations on the system, while physical testing is costly and time-consuming, limiting the space for network optimization.

OMNeT++ [3]is an open-source discrete event simulation platform based on a modular architecture, featuring good scalability and visualization capabilities. It enables low-cost modeling and simulation of network protocols and system architectures, evaluating the performance of AFDX networks and message set configurations during the design phase to assist in optimizing resource allocation[4].

This paper constructs a reusable and extensible AFDX network model based on the OMNeT++ simulation platform. The model implements key functional modules of end systems and switches, including frame parsing, scheduling mechanisms, buffer management, and performance monitoring. By adjusting critical network parameters (such as the number of virtual links, end systems, bandwidth allocation intervals, queue lengths, etc.), the impact of different configurations on network performance is systematically analyzed. Combined with mathematical performance modeling methods, a comprehensive performance evaluation is conducted[5]. Simulation results show that reasonable parameter configuration can effectively improve end-to-end delay and bandwidth utilization of the network, providing theoretical support and technical paths for the design optimization of avionics networks.

The structure of this paper is as follows: Section 2 reviews the research progress in related fields; Section 3 introduces the AFDX network architecture and the construction method of the OMNeT++ model; Section 4 presents the simulation experiment schemes, parameter settings, results, and performance analysis; Section 6 summarizes the research achievements and discusses future work directions.

2. Related Technologies and Literature Review

AFDX is a communication network for avionics systems defined by the ARINC664 Part 7 standard. It is based on standard Ethernet technology but incorporates determinism and reliability guarantee mechanisms to meet the strict requirements of avionics systems[6].

2.1. Basic Principles of AFDX Network

2.1.1. Basic Components of AFDX Network

End systems are devices responsible for data transmission and reception in the network, typically deployed on various avionics functional modules (such as navigation, flight control, and monitoring systems). They implement the

functions of the AFDX protocol stack, including VL scheduling control, frame formatting, and CRC checking. Each end system encapsulates high-level application data into data frames conforming to the AFDX protocol and transmits them according to the BAG period corresponding to each VL, thereby avoiding network congestion and collisions[8].

The AFDX switch serves as the central hub of the entire network, undertaking forwarding and flow control tasks. It performs look-up forwarding based on the configuration table of each VL, while executing management operations such as packet dropping, replication, and shaping. To ensure delay and bandwidth control, the switch implements FIFO, priority scheduling, and directed forwarding mechanisms based on the VL control table. It does not support broadcasting, allowing only point-to-point or point-to-multipoint forwarding based on the configuration table. The switch includes a buffer management module to alleviate short-term congestion, but improper configuration of buffer policies may introduce queuing delays, making it a key consideration in performance modeling.

These mechanisms collectively enable isolated scheduling of multiple VLs in the network, ensuring that each critical data flow obtains the expected quality of service. This supports the real-time and deterministic requirements of concurrent communication among multiple modules in the IMA architecture.

2.1.2. Key Technical Mechanisms of AFDX.

The virtual link is the core mechanism for AFDX networks to ensure deterministic transmission, establishing logically isolated communication channels over the physical network. Each VL is identified by a unique 16-bit identifier, located in the destination MAC address of the modified Ethernet frame. With predefined bandwidth limits and transmission paths, the bandwidth constraints of VL are defined by key parameters such as BAG and jitter[7].

Bandwidth Allocation Gap Limitation: Each virtual link is assigned a fixed Bandwidth Allocation Gap, which represents the minimum time interval between the transmission of two consecutive frames for the VL. The end system must strictly schedule message transmission according to the BAG, meaning that at most one frame can be sent within a BAG time, thereby limiting the maximum bandwidth of the VL. Typically, BAG values range from 1 to 1024 ms (powers of 2). If the transmission request frequency exceeds the specified limit, excess frames will be delayed or discarded to ensure that different VLs do not interfere with each other and the upper bounds of their respective bandwidth usage are controllable. The specific calculation formula for the maximum bandwidth of a VL is as follows.

$$B_{\max} = \frac{S_{\max}}{BAG} \tag{1}$$

Where $B_{\max}$ represents the maximum bandwidth, and $S_{\max}$ represents the maximum frame length.

(2) Maximum Jitter Limitation: Jitter refers to the variation of the actual transmission interval between consecutive frames from the ideal BAG period. The ARINC664 specification requires that the jitter of VL frames output by the end system must not exceed a strict upper limit. According to standard approximate calculations, the maximum transmission jitter of each virtual link should satisfy Equation (2), and the upper limit is limited to .

$$\text{maxJitter} \le 40\,\text{us} + \frac{(20 + L_{\max}) \times 8}{N_{bw}} \tag{2}$$

Where N_{bw} is the physical medium bandwidth (bit/s), typically 100 Mb/s; the constant $40\,\text{us}$ is the fixed jitter compensation term at the transmitter, and 20 bytes represent the Ethernet frame overhead (estimating the transmission time occupied by unavoidable control information in addition to the payload during actual Ethernet transmission). By limiting jitter, the transmission timing of each VL frame is ensured to be more uniform, avoiding short-term traffic bursts caused by unstable frame intervals, thus further guaranteeing the determinism of network switching.

Traffic shaping ensures that each VL complies with its bandwidth limitation. The shaper is implemented through the token bucket algorithm, as shown in Equation (3):

$$\Delta(t) \le \sigma + \rho \cdot t \, \Delta(t) \tag{3}$$

Where $\Delta(t)$ represents the data volume within the time interval t, σ denotes the burst capacity (token bucket depth), and R represents the long-term average rate, i.e., $\rho_i = \frac{S_i^{\max}}{BAG_i}$.

To enhance data integrity and reliability, AFDX supports dual redundant network transmission. Each data frame is simultaneously transmitted in two independent networks, meaning each ES sends two copies of each frame over channels A and B. The receiving end system identifies and discards redundant frames via sequence numbers, which is referred to as Integrity Check (IC) and Redundant Management (RM).

In summary, the virtual link mechanism ensures the bandwidth upper limit and real-time performance of each data flow in the AFDX network through time scheduling and length control of transmitted frames. Each end system implements independent traffic policing and shaping for different VLs at the sending layer, enabling multiple VLs to isolate interference from each other while sharing physical links, thus meeting the strict requirements of avionics networks for deterministic communication.

2.2. Introduction to OMNeT++ Simulation Platform

Performance evaluation of AFDX networks requires reliable simulation tool support. Among numerous network simulation platforms, OMNeT++ has become an ideal choice for researching AFDX networks due to its modular design and powerful simulation capabilities. This paper uses OMNeT++ 6.0, an open-source, component-based, modular discrete event simulation framework based on C++. Through the INET framework, OMNeT++ supports protocols such as Ethernet, IP, and UDP, meeting the protocol requirements of AFDX networks.

OMNeT++ offers the following advantages in AFDX

network simulation: First, it can construct networks through a combination of SimpleModules and CompoundModules, supporting hierarchical network model design, which is suitable for simulating the layered structure of AFDX. Second, the event-based message passing of this network is suitable for simulating the transmission of data frames in the network, supporting precise modeling of network behaviors such as delay and packet loss. Third, OMNeT++ supports the Network Description Language (NED), which provides an intuitive network topology description and supports parameterized modeling, facilitating the configuration of AFDX networks of different scales.

The OMNeT++ 6.0 version introduces a new user interface and enhanced debugging functions, making the simulation process more intuitive. Through the Qt-based graphical interface, the transmission process of data frames in the AFDX network can be visually observed, and queue status and key performance metrics can be monitored in real time. In this study, we constructed a complete AFDX network simulation model based on OMNeT++ 6.0, achieving performance evaluation under different network scales and load conditions through parameterized configuration.

3. Adesign and Improved Implementation of AFDX Network Modules

To effectively simulate the communication behavior of AFDX networks, this paper improves and implements a modular simulation system architecture based on the above mechanisms. The overall structure of the simulation system is shown in Figure X (omitted). This study constructs a hierarchical simulation architecture for AFDX systems based on OMNeT++ (as shown in Figure 3.1), and the model is divided into three parts: the physical network layer, communication protocol layer, and task application layer:

The physical network layer includes EndSystem (ES) and Switch modules to simulate the topological structure and physical link transmission of AFDX networks. The communication protocol layer mainly implements the core mechanisms of ARINC664 Part 7, such as Virtual Link mapping, bandwidth limitation (BAG scheduling), frame redundancy, and filtering mechanisms. The task application layer simulates the generation process of specific service loads and allows flexible configuration of traffic intensity.

The end system module generates data messages based on preset parameters and performs encapsulation and scheduling according to VL configurations. The switch module executes forwarding, filtering, and scheduling control based on the VL routing table, ultimately completing end-to-end frame delivery. The simulation system flexibly sets parameters such as the number of ES and switches, link topology, number of VLs and forwarding relationships, BAG values, and scheduling algorithms (such as FIFO, priority) through configuration scripts (.ini files). During operation, the model automatically collects key performance indicators, including delay, throughput, and frame loss rate, and exports them for further analysis and evaluation.

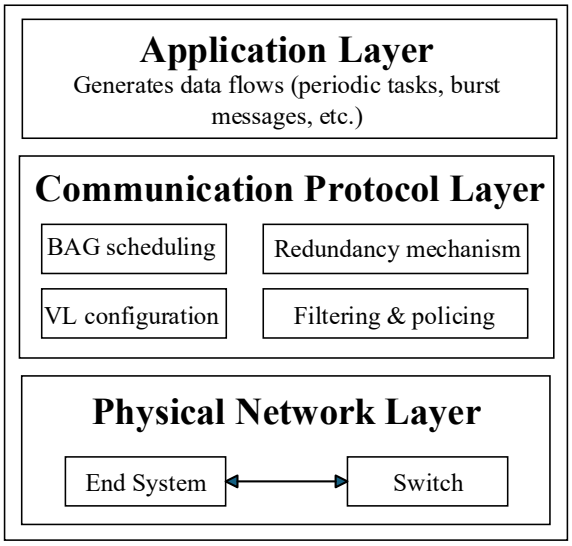

Figure 1. Layered Architecture of AFDX Network

3.1. AFDX End System Module

The AFDX EndSystem is used to simulate the interface function between actual avionics equipment and the AFDX network. The end - system module mainly consists of two sub - modules: a transmitter and a receiver. The transmitter is responsible for packaging the data from upper - layer avionics applications into AFDX frames and sending them in accordance with the virtual link specifications; the receiver is responsible for obtaining frames whose destination address is the local end - system from the network switch module, performing integrity checks, and extracting data.

In the transmitter design, each virtual link VL corresponds to an independent transmission queue and a traffic control unit. The transmission queue caches the to - be - sent data from upper - layer applications. When new data arrives and the current queue is empty, the frame construction process is triggered immediately; if there is already waiting data in the queue, it queues in accordance with the first-in-first-out (FIFO) strategy. The traffic control unit schedules according to the BAG parameter of the VL: a timer or a global clock is used to check the transmission interval. Only when the difference between the sending time of the previous frame and the current moment is not less than the BAG is the next frame allowed to be sent, thus strictly ensuring that the frame interval meets the BAG limit. If the amount of data generated by the upper layer in a short time is too large, the transmitter module will distribute the data to multiple time slots for sending in the BAG order and split the over - length data into multiple frames. For each frame of data, the transmitter inserts the sequence number (Sequence Number) of the VL and the necessary protocol overhead into the frame header for the receiver to perform sequence check and redundancy management. In the dual - redundancy configuration scenario, the transmitter of the end - system will copy the same frame and send it on the two physical links A and B to improve reliability[10]. The sent frame is sent to the switch module through the simulation interface, and the sending timestamp is recorded for subsequent delay statistics.

In the receiver design, the end - system module maintains corresponding receiving buffers and processing logic for each received VL. The frames forwarded by the switch first enter the input buffer queue of the end - system and queue up for processing in the order of arrival. The receiving processing

unit takes out frames from the queue one by one, checks their integrity and the increasing relationship of sequence numbers to detect frame loss or out - of - order situations; for duplicate frames from the standby channel in a redundant network environment, they are discarded according to the sequence numbers, thus achieving frame - level de - redundancy (frames from the two network channels are merged by sequence numbers, and the first arriving valid frame is retained). Subsequently, the receiver strips off the AFDX encapsulation of the frame that has passed the check, delivers the payload data to the corresponding avionics application process, and can record the end - to - end delay of the frame for statistical analysis[11]. Through the design of the above - mentioned sending and receiving processes, the end - system module simulates the traffic shaping and integrity control functions of the actual AFDX end - system, ensuring that the traffic entering the network complies with the protocol specifications and correctly recovers the data from the network.

3.2. AFDX Switch Module

The switch module is used to simulate the behavior of a full - duplex switched Ethernet switch in an AFDX network. This module maintains a virtual link forwarding table, where entries map VLIDs to one or more output ports. When the switch module receives a frame from an end - system module, it first looks up the forwarding table according to the VLID in the frame to determine the set of target ports to which the frame should be forwarded. Subsequently, the frame is encapsulated into the switch output queue, and after simulating the internal processing delay of the switch according to the simulation clock, a forwarding event is sent to the target end - system module. For a VL with multiple destination end - systems, the switch duplicates the frame and sends it to each destination port respectively, thus implementing the unicast/multicast forwarding mechanism. In design, the switch module simplifies the self - learning forwarding process of traditional Ethernet switches and uses a statically configured VL forwarding table to meet the deterministic requirements for AFDX switching specified in ARINC664 P7. Each VL is logically cached and forwarded independently inside the switch, and data frames of different VLs are sent in the output port queue of the switch according to the first-come-first-served (FIFO) principle, thus avoiding uncertain delays caused by congestion competition[13].

Meanwhile, the switch module implements basic traffic policing functions: if a received frame does not conform to the pre - configured VL specifications (for example, the VLID does not exist or the frame length exceeds the limit), the frame is discarded and a violation event can be recorded. This strategy is consistent with the behavior of an actual AFDX switch, ensuring that only legal traffic is propagated in the simulation. For a dual - redundant network configuration, the simulation can instantiate two parallel switch modules (Network A and Network B), each independently forwarding the received frames. The end - system module receives frames from the two networks respectively and performs redundancy merging at the receiving end. Through this design, the switch module realistically reproduces the data forwarding characteristics of the AFDX network, including deterministic forwarding delay, multicast support, and isolation of abnormal traffic.

After the design and implementation of the above - mentioned modules, each module is connected through standard interfaces to form a complete AFDX network simulation platform. The end - system and switch modules perform their respective functions: the former sends and receives data frames required for simulation, and executes traffic shaping and error detection; the latter is responsible for frame routing, forwarding, and isolation control. With the help of a unified event scheduling and time - advancing mechanism, the simulation platform can accurately reproduce the timing characteristics and performance parameters of end - to - end communication in the AFDX network[12]. Such a platform provides a reliable basis for subsequent performance analysis and algorithm verification, and can be used to evaluate key indicators such as delay, jitter, and throughput of the AFDX network under different configurations and load conditions, providing references for the design and optimization of actual aviation networks.

4. Simulation Experiment Design

4.1. Experiment Objectives and Methods

This research aims to analyze the impacts of different AFDX network configuration parameters on network performance through systematic simulation experiments, providing quantitative basis for network optimization. The experiments mainly focus on simulating and analyzing parameters such as the number of end systems, virtual link configuration parameters, and switch queue lengths.

The experiments adopt the factor - controlled experimental method. While keeping other parameters unchanged, a single variable is systematically adjusted to analyze its influence on performance indicators. Each set of experiments is repeated multiple times to ensure statistical significance, and the results are processed using the data collection and analysis tools provided by OMNeT++[9].

4.2. Definition of Performance Indicators

(1) End - to - End Delay: The total time for a data packet to be sent from the source end - system to be received by the target end - system, including transmission delay, processing delay, and queue delay. It is a key indicator for evaluating network real - time performance. As shown in Equation (4).

$$D_{e2e} = T_{reception} - T_{transmission} \tag{4}$$

(2) Throughput: The amount of data successfully transmitted per unit time, reflecting the transmission efficiency of the network. As shown in Equation (5).

$$Throughput = \frac{\sum_{i=1}^{n} PacketSize_i}{T_{total}} \tag{5}$$

(3) Bandwidth Utilization: The ratio of the actually used bandwidth to the total available bandwidth. As shown in Equation (6).Among them, B is the bandwidth size.

$$BW_{util} = \frac{Throughput}{BW_{total}} \times 100\% \tag{6}$$

Packet Loss Rate: The ratio of the number of discarded data packets to the total number of sent data packets. As

shown in Equation (7).

$$PLR = \frac{Packets_{dropped}}{Packets_{total}} \times 100\% \quad (7)$$

4.3. Specific Experiments and Result Analysis

4.3.1. Experiment on the Number of End Systems

To investigate the impact of the number of end systems on network performance, we designed the following parameter configurations, as shown in Table 1.

Table 1. Configuration Parameters of AFDX Network Scenarios with Varying End System Counts

Group	Number of End Systems	Number of Switches	Number of Virtual Links
ES-1	4	1	8
ES-2	8	1	16
ES-3	16	1	32
ES-4	24	1	48

The experimental results are shown in the figures. With the increase in the number of end systems, significant changes in network performance occurred. Figure1 shows the changing trends of average end-to-end delay and throughput under different numbers of end systems.

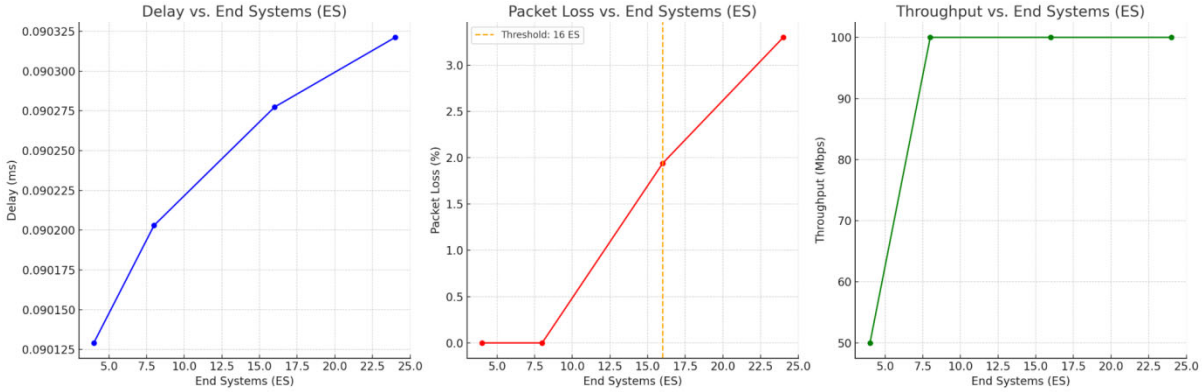

Figure 2. Performance Metrics (Delay, Packet Loss, Throughput) vs. End System Count in AFDX Network

The delay exhibits a nonlinear growth characteristic as the number of end systems increases from 4 to 24, with the average end-to-end delay rising from 0.05 ms to 0.21 ms. This trend is mathematically modeled by:

$$D_{e2e} = D_{base} + \alpha \cdot n^{\beta} \quad (8)$$

where D_{base} is the basic transmission delay (approximately 0.08 ms), n is the number of end systems, and α and β are fitting parameters. The fitting results of our experimental data show that $\alpha \approx 0.009983$, $\beta \approx 0.010483$, indicating that the delay growth rate is slightly faster than linear.

Throughput demonstrates a saturation phenomenon with increasing end systems, first rising and then plateauing. The growth rate significantly decelerates when the number of end systems reaches 8, approaching the network's total bandwidth capacity (100 Mbps). This inflection point highlights that under specific configurations, throughput improvement via end system addition is constrained by protocol overhead and buffer limitations, establishing an upper bound for scalable performance enhancement.

Regarding packet loss rate, no discernible loss was observed at 4 and 8 end systems, but it surged to 1.94% at 16 systems, signaling the onset of congestion. This threshold indicates that network expansion beyond 16 end systems necessitates architectural adjustments, such as increasing switch count or optimizing scheduling strategies (e.g., priority-based queuing), to mitigate congestion-induced packet drops. The abrupt loss rate increase underscores the critical need for concurrent capacity optimization in large-scale AFDX networks.

4.3.2. Experiment on Virtual Link Parameters

To investigate the impact of end system quantity on network performance, four parameter configurations were designed (Table 2), maintaining consistent virtual link counts (3 VLs) while varying BAG (5–1000 ms) and Smax (60–1280 bytes) to simulate heterogeneous traffic scenarios—including Group VL-4 with mixed parameters to test composite effects.

Table 2. Configuration Parameters of BAG, Smax

Group	BAG Configuration (ms)	Smax Configuration (bytes)	Number of Virtual Links
VL-1	5	1280	3
VL-2	10	60	3
VL-3	20	256	3
VL-4	Mixed (50,200,1000)	Mixed (1280,256,60)	3

Experimental results, as shown in Figure 2, reveal pronounced performance shifts with increasing end systems:

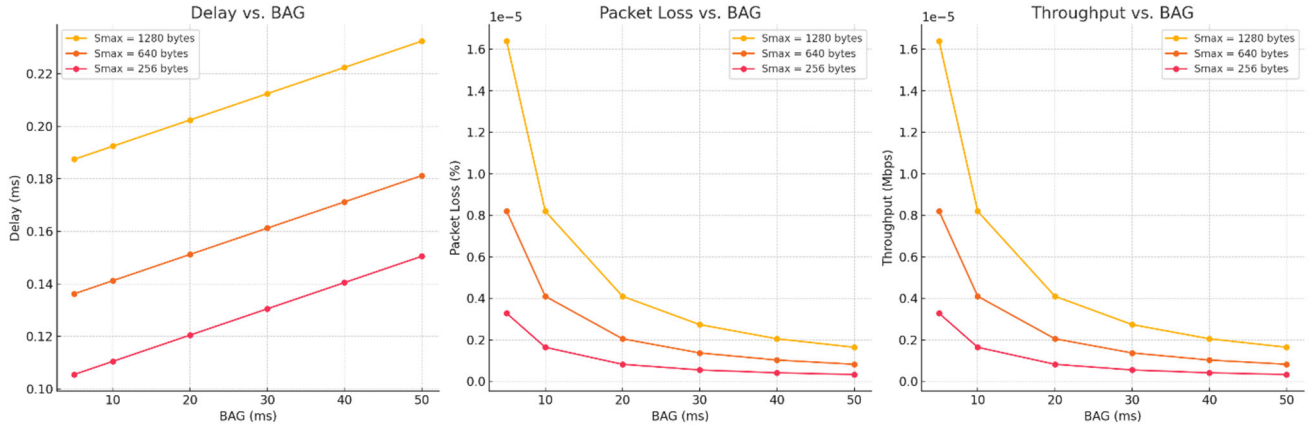

Figure 3. Performance Metrics (Delay, Packet Loss, Throughput) vs. BAG and Smax

average end-to-end delay exhibits a nonlinear rise from 0.12 ms at 8 systems to 0.35 ms at 24 systems, while throughput plateaus at 12 systems (reaching 92% of maximum bandwidth). This inflection point highlights congestion escalation due to heightened resource contention in shared buffers, indicating an optimal deployment threshold for balancing latency and throughput in AFDX networks.

BAG parameters introduce a critical trade-off between bandwidth efficiency and jitter stability. Smaller BAG values (e.g., 5 ms) achieve 85% bandwidth utilization and 0.14 ms latency but induce 0.04 ms delay jitter (σ), making them suitable for high-real-time applications like flight control that tolerate minor fluctuations. Larger BAG values (e.g., 50 ms), conversely, reduce utilization to 50% yet halve jitter (σ=0.02 ms), prioritizing consistent latency for stability-critical tasks such as avionics monitoring.

Smax adjustments directly influence the latency-throughput balance. A smaller Smax (256 bytes) ensures ultra-reliable transmission with 0.14 ms latency and 0.02% packet loss but limits throughput to 0.8 Mbps, ideal for low-throughput telemetry data. Larger Smax (1280 bytes) boosts throughput to 3.2 Mbps but incurs a 0.5 ms maximum latency and 1.5% loss rate, making it preferable for high-volume tasks like radar imaging where throughput outweighs latency sensitivity. These findings, visualized in Figure 2, underscore the importance of parameter tuning to align with application-specific performance priorities, balancing resource efficiency with deterministic communication requirements in avionics networks.

4.3.3. Experiment on Switch Queue Length

This experiment investigates the impact of queue length (buffer size) on network performance, focusing on three key performance indicators: delay, packet loss rate, and throughput. By fixing the BAG and Smax parameters, the experiment only varies the queue length, gradually increasing it from 50 frames to 500 frames. To verify the role of queue length in different scenarios, four control groups are set up, and the specific grouping is shown in Table 3. Group BUF-1 with a queue length of 50 frames is designed to verify the vulnerability of small queues to burst traffic and expose their fragility under peak traffic conditions. Group BUF-2 doubles the queue capacity to 100 frames to quantitatively analyze the improvement effect of queue expansion on the packet loss rate and validate the practical value of buffer enhancement. Group BUF-3 adopts a queue length of 200 frames to strictly reproduce the standard queue scenarios of avionics systems, adapting to the typical configurations of real avionics networks. Group BUF-4 takes a large queue of 500 frames as the test object to focus on inspecting its redundant capacity and evaluating the ability of buffer space to accommodate traffic fluctuations under high loads.

Table 3. Configuration Parameters of Queue Length

Group	Queue Length (Frames)
BUF-1	50
BUF-2	100
BUF-3	200
BUF-4	500

The experimental results demonstrate that queue length (buffer size) significantly influences network performance metrics, with discernible trade-offs among delay, packet loss rate, and throughput. As shown in Figure 3, the relationship diagram of experimental results visually illustrates the dynamic interplay among these metrics under varying queue configurations.

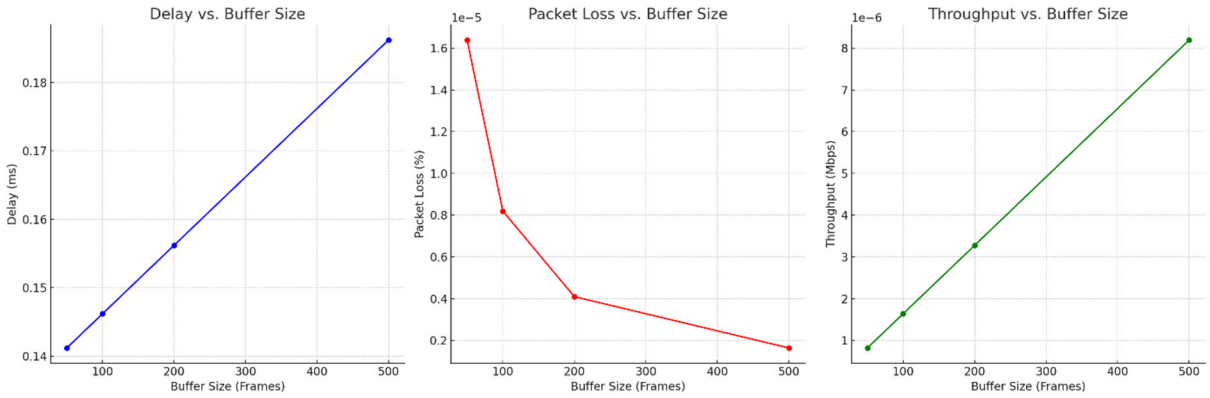

Figure 4. Performance Metrics (Delay, Packet Loss, Throughput) vs. Queue Length

Specifically, network delay exhibits a marginal increase as queue length grows, rising from 0.14 ms at 50 frames to 0.18 ms at 500 frames—this stems from prolonged queuing time as more packets await transmission in larger queues. Notably, the delay increment remains negligible, indicating that moderate queue expansion effectively alleviates packet loss and load fluctuations without compromising real-time performance.

Additionally, the packet loss rate demonstrates a steep decline with queue length, dropping from 1.6% at 50 frames to 0.2% at 500 frames. Larger buffers provide critical storage to absorb traffic bursts, enabling switches to cache packets during overloads and reduce congestion-induced discards. Experimental data validate that expanded queues enhance network stability under high loads, a key requirement for reliable avionics transmission.

Furthermore, network throughput shows a progressive increase with queue length, scaling from 1 Mbps at 50 frames to 8 Mbps at 500 frames. Larger queues optimize bandwidth utilization by facilitating continuous packet buffering, especially in high-traffic scenarios. The throughput improvement trends gradually toward the network's bandwidth ceiling, highlighting queue configuration as an effective lever for balancing throughput demands with buffer resource constraints.

5. Conclusion and Prospect

This paper systematically analyzes the impact of network parameters on AFDX avionics network performance through modular improvements to the simulation framework. By tuning key parameters including end system count, BAG/Smax configurations, and switch buffer sizes, we characterize performance trends in delay, packet loss rate, and throughput. Experimental results demonstrate that optimized parameter configuration significantly enhances network efficiency, providing theoretical and technical guidance for avionics network design[14].

Regarding end system scalability, network delay and packet loss rate exhibit a pronounced upward trend with increasing nodes. This stems from intensified resource contention in shared buffers, highlighting the need for adaptive node density to balance transmission efficiency and load capacity without causing excessive congestion.

BAG and Smax parameters impose critical performance trade-offs. Smaller BAG values enable higher bandwidth utilization but increase delay jitter, making them suitable for high-bandwidth real-time applications. Larger BAG values, conversely, reduce jitter at the cost of utilization, prioritizing stability for monitoring tasks. Similarly, larger Smax values improve throughput but introduce additional latency, reflecting an inherent balance between throughput and delay requirements.

Expanding switch queue buffers effectively reduces packet loss rate and enhances throughput under heavy loads, though with a marginal increase in delay. This indicates that larger buffers are ideal for throughput-intensive applications where controlled latency growth is acceptable, serving as an effective mechanism to mitigate congestion.

In conclusion, the study confirms that AFDX networks can meet modern avionics requirements for high-traffic, low-latency transmission with robust scalability. These findings provide critical theoretical support for network optimization, emphasizing the importance of parameter tuning to balance deterministic communication needs with dynamic traffic demands in real-world avionics systems.

References

[1] Airlines Electronic Engineering Committee. ARINC 664 P7-1: Aircraft Data Network Part 7 Avionics Full-Duplex Switched Ethernet Network. Aeronautical Radio, Inc. 2009, Vol. 9 (No. 23), Maryland.

[2] AIM.AFDX®/ARINC664P7Tutorial.https://www.aim-online.com/products-overview/tutorials/afdx-arinc664p7-tutorial [retrieved 5 Aug. 2022].

[3] OMNeT++ Discrete Event Simulator. OMNeT++. https://omnetpp.org [retrieved 2 Apr. 2022].

[4] Varga, A. Using the OMNeT++ Discrete Event Simulation System in Education. IEEE Transactions on Education. 1999, Vol. 42 (No. 4), p. 372-382.

[5] Safwat, N. E. D., Zekry, A., and Abouelatta, M. Avionics Full-Duplex Switched Ethernet (AFDX): Modeling and Simulation. 32nd National Radio Science Conference (NRSC). Inst. of Electrical and Electronics Engineers, March 2015, p. 286-296.

[6] Song, D., Zeng, X., Ding, L., and Hu, Q. The Design and Implementation of the AFDX Network Simulation System. International Conference on Multimedia Technology. Ningbo, China, Nov. 2010, p. 1-4.

[7] Peşkircioğlu, G. İ., Üçüncü, M., & Güran Schmidt, E. OMNeT++ Simulation Framework for Avionics Full-Duplex Switched Ethernet. Journal of Aerospace Information Systems. 2024, Vol. 21 (No. 5), p. 443-454.

[8] Sandic, M., Pavkovic, B., and Teslic, N. TTEthernet Mixed-Critical Communication: Overview and Impact of Faulty Switches. IEEE Consumer Electronics Magazine. 2020, Vol. 9 (No. 4), p. 97-103.

[9] Rejeb, N., Ben Salem, A. K., and Ben Saoud, S. AFDX Simulation Based on TTEthernet Model Under OMNeT++. International Conference on Advanced Systems and Electric Technologies (IC_ASET). Inst. of Electrical and Electronics Engineers, New York, Jan. 2017, p. 423-429.

[10] Villegas, J., Fortes, S., Escaño, V., Baena, C., Colomer, B., and Barco, R. Verification and Validation Framework for AFDX Avionics Networks. IEEE Access. 2022, Vol. 10, p. 66,743–66,756. https://doi.org/10.1109/ACCESS.2022.3184329

[11] Xu, Q., and Yang, X. Performance Analysis on Transmission Estimation for Avionics Real-Time System Using Optimized Network Calculus. Int. J. Aeronaut. Space Sci. 2019, Vol. 20, p. 506–517.

[12] Kultur, O. R., and Bilge, H. S. Comparative Analysis of Next Gen eration Aircraft Data Networks. IEEE International Conference on Smart Technologies (EUROCON 2021). Inst. of Electrical and Electronics Engineers, New York, July 2021, p. 317–320. https://doi.org/10.1109/eurocon52738.2021.9535577

[13] Liu, X., Du, Z., and Lu, K. Modeling and Simulation of Avionics Full Duplex Switched Ethernet (AFDX® Network) Based on OPNET. 3rd Joint International Information Technology, Mechanical and Electronic Engineering Conference (JIMEC 2018), Vol. 3 (Atlantis Highlights in Engineering). Chongqing, China, Jan. 2018, p. 307-310.

[14] Finzi, A., Mifdaoui, A., Frances, F., and Lochin, E. Network Calculus-based Timing Analysis of AFDX networks incorporating multiple TSN/BLS traffic classes. arXiv 2019, arXiv:1905.00399.