Received 19 June 2023; revised 14 August 2023; accepted 29 August 2023. Date of publication 5 September 2023; date of current version 3 October 2023.

Digital Object Identifier 10.1109/OJCOMS.2023.3312110

Rapid Prototyping and International Validation Activity for the L-Band Digital Aeronautical Communications System (LDACS)

KAZUYUKI MORIOKA[1] (Member, IEEE), AKIKO KOHMURA[1] (Member, IEEE), NARUTO YONEMOTO[1] (Member, IEEE), LEONARDUS J. A. JANSEN[2], NILS MÄURER[2] (Member, IEEE), THOMAS GRÄUPL[2], AND MICHAEL SCHNELL[2] (Senior Member, IEEE)

[1]Electronic Navigation Research Institute, National Institute of Maritime, Port and Aviation Technology, Chofu 182-0012, Japan
[2]Institute of Communications and Navigation, German Aerospace Center, 82234 Wessling, Germany

CORRESPONDING AUTHOR: K. MORIOKA (e-mail: morioka@mpat.go.jp)

This work was supported in part by JSPS KAKENHI under Grant 20K19795.

ABSTRACT The L-band Digital Aeronautical Communications System (LDACS) is a cellular-based broadband, secure digital aeronautical communications system designed to enhance the safety and efficiency of air traffic management (ATM) through the facilitation of innovative ATM paradigms, such as 4D trajectory-based operations (TBO). LDACS is in its final stages of standardization by the International Civil Aviation Organization (ICAO) and the European Organisation for Civil Aviation Equipment (EUROCAE). Furthermore, it has been introduced to not only the aviation industry but also the Internet community in the form of an informational Request For Comments (RFC). Rapidly creating prototypes of an international communications standard by multiple organizations is crucial to the effective deployment of that standard. This approach enables validation and interoperability testing across various countries' prototypes. Therefore, first, we create an LDACS prototype through a software/hardware co-design strategy by Software Defined Radio (SDR) and High-Level Synthesis (HLS). This approach expedites and economically streamlines the development process. Second, we show the alignment of our prototype with the LDACS specification through preliminary and cell entry tests. Third, we demonstrate the efficacy of LDACS' Quality of Service (QoS) and security features via end-to-end IPv6 connectivity and security tests. Finally, the soundness and clarity of the LDACS specification is evidenced via interoperability tests between our prototype and a European counterpart.

INDEX TERMS Next generation aeronautical communications system, L-band digital aeronautical communications system (LDACS), international civil aviation organization (ICAO), european organisation for civil aviation equipment (EUROCAE), international standardization and validation, rapid prototyping, software defined radio (SDR), high-level synthesis (HLS)

I. INTRODUCTION

THE INTERNATIONAL Air Transport Association (IATA) reports that global air traffic demand has been rebounding from the COVID-19 pandemic and is steadily increasing beyond pre-pandemic levels [1]. To achieve a safer and more efficient air traffic management (ATM), it is crucial to modernize the aeronautical communications systems and introduce new concepts, such as 4D Trajectory-Based Operations (TBO) [2]. This is particularly important since the current air traffic management system relies on Very High Frequency (VHF) voice/data communication, which is struggling to keep up with the increasing saturation of the VHF band in congested airspace.

The LDACS [3], [4] is a cellular-based, broadband, and secure digital aeronautical communications system which is expected to first complement the VHF Data Link Mode 2

(VDL Mode 2) and finally to replace it. LDACS supports digital voice, Internet Protocol Suite (IPS) traffic and enables multiple new applications, such as 4D TBO and System Wide Information Management (SWIM) [5]. It is also expected to be a true integrated-Communications, Navigation and Surveillance (I-CNS) system supporting Ground Based Augmentation System (GBAS) [6], Alternative Positioning Navigation and Timing (APNT) [7], and surveillance capabilities.

LDACS is in its final stages of standardization within the International Civil Aviation Organization (ICAO) and the European Organisation for Civil Aviation Equipment (EUROCAE). Further, LDACS has been introduced to not only the aviation but also the Internet community in the form of an informational Request For Comments (RFC) [8].

Prototyping and validation testing of a new communications system is important to confirm the feasibility of the newly developed standard. To support the LDACS standardization in ICAO and EUROCAE, several prototypes of LDACS have been developed and tested [4], [9], [10]. Furthermore, from the viewpoint of international standardization, interoperability testing is crucial for eliminating ambiguity in standardization documents. This process minimizes confusion and potential implementation errors, ensuring that the various international implementations of the communication system work seamlessly and efficiently with each other.

The purpose of this work is to rapidly develop an LDACS prototype and to conduct validation and interoperability testing to support the standardization of LDACS. In this paper, we introduce the LDACS prototype developed by the Electronic Navigation Research Institute (ENRI) in Japan, and the international validation activities conducted by the ENRI with the German Aerospace Center (DLR).

The rest of this paper is organized as follows: Section II introduces related works on LDACS and its prototyping and testing. Section III gives an overview of the LDACS specification to understand the chosen design of the following sections. Section IV provides a detailed description of the LDACS prototype we developed. Section V presents the international validation activities and the results. Finally, Section VI concludes this paper.

II. RELATED WORK

There is a growing interest in LDACS, especially in Europe, since it is expected to be an alternative to the VHF communications, which are extremely congested in Europe. Some LDACS prototypes that are compatible with the LDACS air-to-ground (A/G) specification [11] were developed and the technical validations were conducted in the Single European Sky Air Traffic Management Research (SESAR) program [9], [10]. In addition, the flight trial was conducted to demonstrate Air Traffic Services (ATS)-Baseline 2 (B2), Internet Protocol version 6 (IPv6) connectivity, and seamless mobility, recently [12].

LDACS is anticipated to encompass not just communication capabilities but also navigation and surveillance functionalities, positioning it as an I-CNS system. The navigation functions of LDACS were validated in Migration towards Integrated COM/NAV Avionics (MICONAV), a German national project. In the MICONAV project, the LDACS-based ranging [13] and the transmission of the GBAS signals over LDACS [14] have been reported. Additionally, these functions were demonstrated in flight trials [4], [15]. Further, a concept of LDACS-based Wide Area Multilateration (WAM) is proposed as one of the surveillance functions [16].

Security is an essential part of the aeronautical communications system. The security architecture of LDACS was proposed and discussed in [17], [18], [19], then the implementation and evaluation were conducted in [20], [21]. However, the evaluations were conducted by computer simulation. To evaluate the security functions in a more realistic environment, security validation activities with the prototype introduced in this paper have been conducted and presented in [22].

For the practical integration of LDACS into the aviation industry, it is important to consider the interference between LDACS and other L-band aeronautical systems. The interference between LDACS and Distance Measuring Equipment (DME) was considered and the cell planning in Europe was studied in [23]. In addition, compatibility testing between LDACS and a Joint Tactical Information Distribution System (JTIDS) was performed in [24].

To mitigate the interference to other L-band aeronautical systems, some variants of LDACS, which improve the spectrum characteristics of Orthogonal Frequency Division Multiplexing (OFDM), were proposed. These include a re-configurable filtered-OFDM [25], variable filtered-OFDM [26], and Filter Bank Multi-Carrier (FBMC) [27]. However, these variants have not been incorporated in the LDACS specification [11] yet.

From a different perspective, receiver structures that utilize nonlinear operations to mitigate interference from DME have been proposed. These include techniques such as deep clipping and joint clipping blanking [28], genie-aided estimator, and optimum Bayesian estimator [29]. Although the LDACS variants [25], [26], [27] affect the LDACS specification, these methods only necessitate an additional processing module at the receiver side and do not require any modifications to the LDACS specification itself.

Rapid prototyping of a new radio system is important to verify the validity and feasibility of the proposed specifications as early as possible. Generally, Software Defined Radios (SDRs) are used for prototyping recent digital radio systems. Universal Software Radio Peripheral (USRP) by Ettus Research$^{\mathrm{TM}}$and National Instruments Corp., [30] is popular SDR platform for research purposes. LDACS prototyping using the USRP was proposed in [31]. However, this work focused only on physical layer implementation; layer 2 protocols were not implemented. Moreover, real-time processing was not considered in their prototype.

In this work, we consider a full protocol stack implementation including not only the physical layer but also layer 2 and layer 3 to realize real-time IPv6 end-to-end communication. To achieve this goal fast and at a low cost, we took a software/hardware co-design approach using a Multiprocessor System-on-Chip (MPSoC) which combines multiple processors, a Field Programmable Gate Array (FPGA), memory, I/O interfaces, and other peripheral devices on a single chip.

The software/hardware co-design using MPSoC was presented in [32]. However, the implementation is physical layer only, and real-time processing was not considered. In [33], a practical LDACS prototype was developed which considers end-to-end real-time communications using MPSoC. This very prototype was used to conduct the interoperability tests in Section V in this paper.

The design approaches taken in [33] and our prototype are similar. One of the differences is that their implementation is based on Linux OS, whereas ours is based on bare-metal (no-OS) development to ensure time-critical operations. Another difference is that they used Hardware Description Language (HDL) to implement the physical layer signal processing, whereas we used the C programming language with High Level Synthesis (HLS) to reduce development costs. Our detailed design is described in Section IV in this paper. Furthermore, detailed discussions and comparisons are presented in Section IV-E.

This paper represents an expanded edition of our previous conference paper [34], encompassing several notable enhancements. Firstly, we have included comprehensive explanations of the LDACS specification in Section III. Additionally, we have incorporated a more detailed description of our software/hardware co-design along with implementation results in Section IV, specifically in Sections IV-C and IV-D. Furthermore, we have introduced international validation activities carried out through collaborative efforts between ENRI and DLR in 2022. These activities encompassed end-to-end tests, security tests, and interoperability tests, which are detailed in Section V, specifically in Sections V-C–V-E, respectively.

III. OVERVIEW OF LDACS SPECIFICATION

In this section, we present a concise overview of the LDACS specification [11] to facilitate our discussion on prototype design in the following section.

A. PHYSICAL LAYER

Communication from a Ground Station (GS) to an Airborne Station (AS) is called a Forward Link (FL), whereas that from an AS to a GS is called a Reverse Link (RL). LDACS uses a fixed radio frame structure that synchronizes all communication participants. Fig. 1 shows the LDACS frame structure.

As shown in the figure, the frame structure consists of a SF of 240 ms duration, which is split into four MFs, each lasting 58.32 ms. The FL frame structure begins with the BC slot where the GS announces the LDACS cell's existence, followed by the Common Control (CC) slot, where resources are assigned to an AS, and the FL DATA frame, where the GS can directly transmit FL user-data.

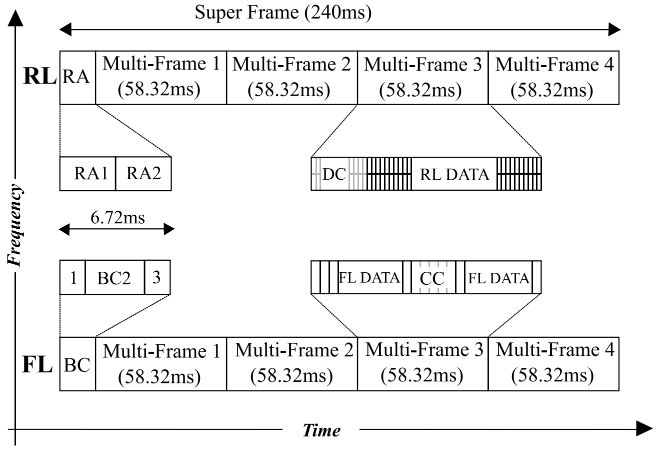


FIGURE 1. LDACS frame structure. The frame structure comprises a SF that lasts for 240 ms. The SF is further divided into 6.72 ms RA/BC slots and four MFs, with each MF lasting for 58.32 ms.

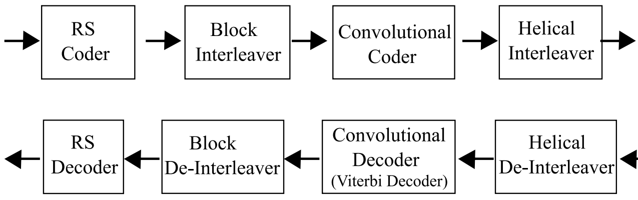


FIGURE 2. Channel coding and interleaving. On the RX side, the complementary operation of the TX side is applied in reverse order. Typically, a Viterbi decoder is used for decoding the convolutional codes.

The RL frame structure of LDACS includes one RA frame at the beginning of each SF, which is split into two slots, where the ASs can request access to the LDACS cell. Both the Dedicated Control (DC) slot, where the ASs must request resources and the RL DATA frame, where the ASs can transmit RL user data after being granted resources by the GS in the CC, occur in every MF.

Fig. 2 illustrates channel coding and block interleavers. LDACS uses a concatenated coding scheme consisting of an outer Reed-Solomon (RS) code and an inner variable-rate convolutional code as its Forward Error Correction (FEC) scheme. At the TX side, the information bits are scrambled with a fixed randomizer pattern before being processed by the RS encoder. The block interleaver is then applied to the encoded bits, followed by zero-terminating convolutional coding. In the final step, the coded bits are interleaved using a helical interleaver. At the RX side, the complementary operation is applied in the reverse order. Typically, a Viterbi decoder [35], [36], which is an efficient decoding algorithm for convolutional codes, is used on the RX side.

LDACS utilizes an adaptive Coding and Modulation Scheme (CMS) that depends on the channel quality, providing user data rates of 230.53 to 1428.27 kbps on the FL and 235.3 to 1390.40 kbps on the RL per operated cell.

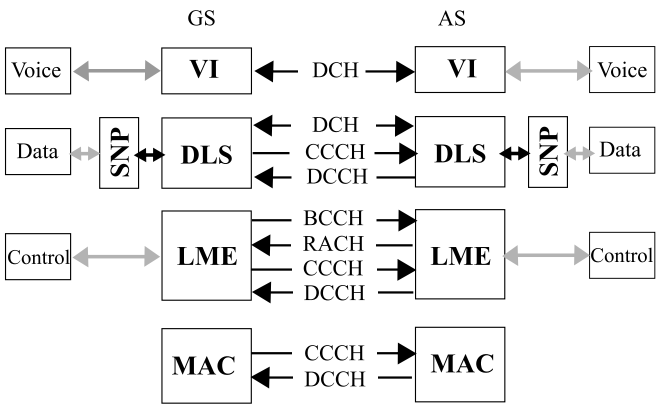


FIGURE 3. The LDACS logical channel structure in AS and GS. The LDACS layer 2 protocol consists of VI, SNP, DLS, LME, and MAC entities. The figure illustrates the mapping of the logical channels utilized by these entities.

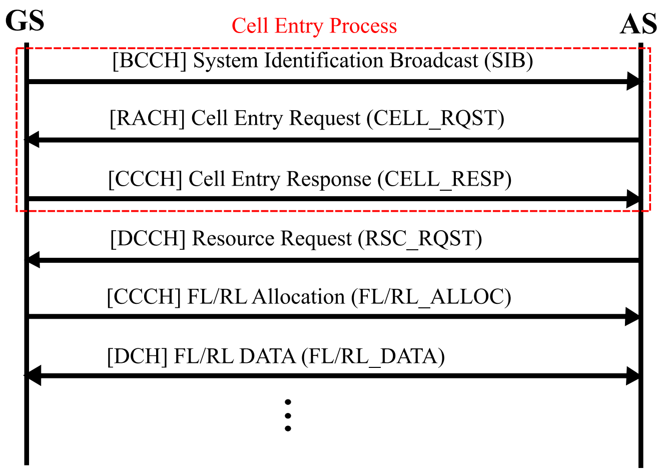

GS
Cell Entry Process
AS
[BCCH] System Identification Broadcast (SIB)
[RACH] Cell Entry Request (CELL_RQST)
[CCCH] Cell Entry Response (CELL_RESP)
[DCCH] Resource Request (RSC_RQST)
[CCCH] FL/RL Allocation (FL/RL_ALLOC)
[DCH] FL/RL DATA (FL/RL_DATA)

FIGURE 4. LDACS protocol sequence. The red square marks the cell entry process, which is followed by resource management and the exchange of user data.

B. LAYER 2 PROTOCOLS

Fig. 3 shows the LDACS logical channel structure in AS and GS. The Voice Interface (VI) supports virtual voice circuits and emulates the traditional party-line voice service using a "listen-before-push-to-talk" approach.

The Sub-Network Protocol (SNP) establishes connectivity between the AS and the GS within the LDACS sub-network and ensures secure communications by employing cryptographic measures. Additionally, it compresses IPv6 headers, verifies data integrity, and routes packets based on their class of service.

The Data Link Service (DLS) facilitates the bi-directional exchange of user data, with options for acknowledged and unacknowledged transmissions. With the acknowledged data link service, the sending DLS entity waits for an acknowledgment from the receiver. If no acknowledgment is received within a specified time, the sender automatically tries to re-transmit its data. Furthermore, the DLS features include segmentation and reassembly functionalities by taking into account the correlation between the size of the Service Data Unit (SDU) and the allocated resource size.

The LDACS Management Entity (LME) provides both mobility and resource management services. The mobility management service supports cell registration and deregistration (cell entry and cell exit), scans Radio Frequency (RF) channels of neighboring cells, and facilitates handovers between cells. The resource management service maintains links by making power, frequency, and time adjustments, and supports Adaptive Coding and Modulation (ACM) as well as resource allocation.

The MAC provides the frame structure necessary to realize slot-based Time Division Multiplex (TDM) access on the physical link. It offers functions for synchronizing the MAC framing structure and the physical layer framing and respectively maps each logical channel, i.e., Broadcast Control CHannel (BCCH), Random-Access Channel (RACH), Common Control CHannel (CCCH), Dedicated Control CHannel (DCCH) and Data CHannel (DCH) to the BC, RA, CC, DC, and FL/RL DATA physical slots.

C. PROTOCOL SEQUENCE

Fig. 4 illustrates the LDACS protocol sequence. First, the AS synchronizes its time and frequency with the synchronization symbols of the BCCH transmitted by the GS. Once synchronization is completed, the AS decodes the System Identification Broadcast (SIB) message to obtain several parameters required for the cell entry process. Then, the AS sends a Cell Entry Request (CELL_RQST) message in the RACH, including the Unique Address (UA) to identify the aircraft.

After the GS received the CELL_RQST message and confirmed the UA to be valid, it sends the Cell Entry Response (CELL_RESP) message and with it assigns the temporary Sub-net Access Code (SAC) address to the AS. In addition, information for correcting the frequency, timing, and power differences between the GS and AS, i.e., Frequency Adaptation Value (FAV), Time Advance Value (TAV), and Power Adaptation Value (PAV) are included in the CELL_RESP message.

When the AS receives the CELL_RESP message, it corrects the frequency, timing, and power offsets and moves to the connected state. Thereafter, the upper layer user data can be communicated between the GS and AS. Note that we do not support security functions here, yet. The detailed secure cell entry process is provided in [21].

Finally, before the AS sends user data, the AS requests resources via a Resource Request (RSC_RQST) message through the DCCH. Then, the GS allocates RL radio resources to the AS by sending an RL allocation (RL_ALLOC) message using the CCCH. The AS then sends the user data using the allocated RL DCH. Similarly, the GS announces FL radio resources to the AS by sending an FL allocation (FL_ALLOC) message using the CCCH. The AS decodes the allocated FL DCH and delivers it to the upper layer.

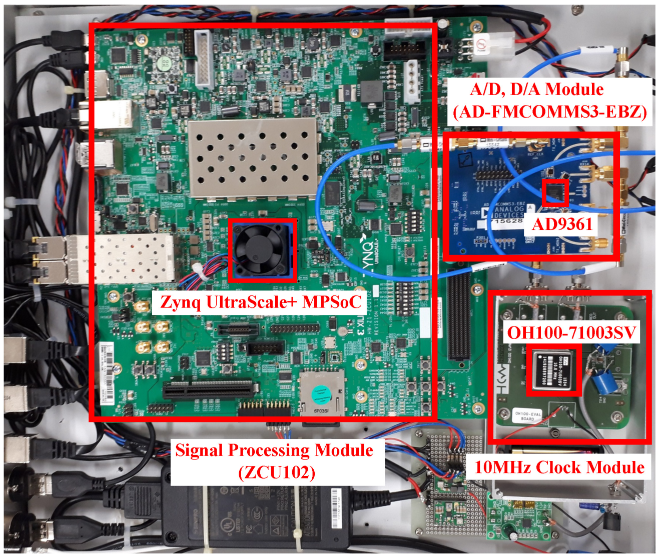


FIGURE 5. The hardware architecture of our prototype. The prototype mainly comprises the AD/DA module, the 10 MHz clock module, and the signal processing module. Each module is commercially available.

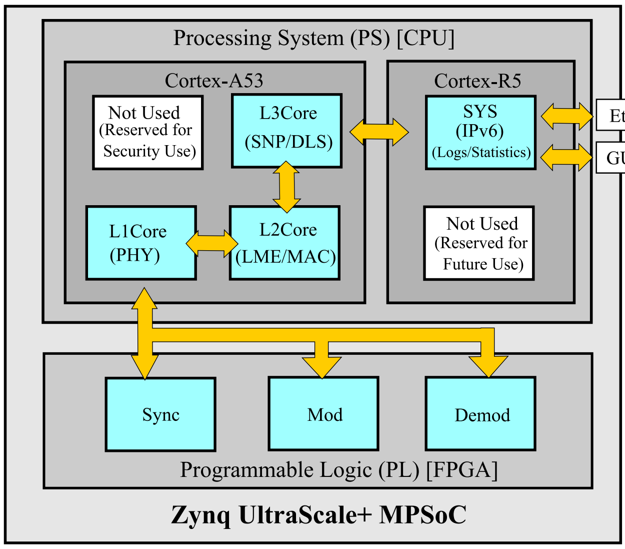


FIGURE 6. The software architecture of our prototype. All modules are implemented in MPSoC. The L1, L2, and L3 cores are in the quad-core A53 processors, and the SYS core is in a dual-core R5 processor of the PS. Synchronization, modulation, and demodulation functions are implemented in the PL.

IV. RAPID PROTOTYPING

In this section, we present our prototype design in detail.

A. HARDWARE ARCHITECTURE

Fig. 5 shows the hardware architecture of our prototype. A prototype consists of an AD/DA module, a clock module, and a signal-processing module.

To ensure compatibility with various aeronautical radio systems, the AD/DA module should have a broad frequency range and the ability to handle a wide instantaneous bandwidth. Therefore, we selected the AD-FMCOMMS3-EBZ, an evaluation board of AD9361 by Analog Devices Inc. The AD9361 supports a frequency band of up to 6 GHz and up to an instantaneous bandwidth of 56 MHz, which cover most aeronautical radio systems. It also supports Multiple-Input Multiple-Output (MIMO) technology. Thus, it can be used to study aeronautical communications systems that employ multiple antennas.

An external clock module is necessary because the frequency stability of the onboard crystal oscillator on the AD-FMCOMMS3-EBZ is not sufficient to support Air/Ground (A/G) radio synchronization. Therefore, OH100-71003SV, an Oven-Controlled Crystal Oscillator (OCXO) by Connor-Winfield Corp., which has a frequency stability of ±10 ppb, was selected as an external clock module.

Further, to support real-time communication, the signal processing module must support real-time signal processing and parallel operation of multiple protocol stacks. Therefore, the ZCU102, an evaluation board of the Zynq UltraScale+ MPSoC by Xilinx Inc., was selected. The MPSoC employs both, an FPGA, which is suitable for processing simple and iterative calculations, and a Central Processing Unit (CPU), which is suitable for implementing complicated protocol behavior, on a single chip. The CPU and FPGA parts in the Xilinx MPSoC are called Processing System (PS) and Programmable Logic (PL), respectively.

The PS comprises a quad-core Cortex-A53 CPU (A53) and a dual-core Cortex-R5 CPU (R5). The frequencies of the A53 and R5 cores are 1.5 GHz and 600 MHz, respectively. The A53 is a 64-bit system, whereas the R5 is a 32-bit system. The PL consists of a middle-scale FPGA which includes 912 Block RAM (BRAM), 548,160 Flip-Flops (FFs), 274,080 Look Up Tables (LUTs) and 2,520 Digital Signal Processor (DSP) slices.

One benefit of this architecture is the seamless cooperation between the PS and PL. The software architecture and software/hardware co-design approach are shown in the following sub-sections.

B. SOFTWARE ARCHITECTURE

Fig. 6 shows the software architecture of our prototype. As described in Section IV-A, the Zynq UltraScale+ MPSoC was used as the signal processing module. The LDACS protocol stack was implemented utilizing three cores of the quad-core processor A53, namely, L1, L2, and L3 cores and each core works in parallel to realize real-time communication.

The L1 core is responsible for the physical layer protocol and controlling the PL; the L2 core is responsible for the MAC and the LME protocol; the L3 core is responsible for the DLS and SNP protocol. The fourth A53 core is reserved for future use, especially for security implementations.

The System (SYS) core is implemented in one core of the dual-core R5 processor. It is responsible for IPv6 and interfacing between the A53 and R5. It collects logs/statistics for debugging purposes and is also interfacing with the Ethernet and Graphical User Interface (GUI). Here, the

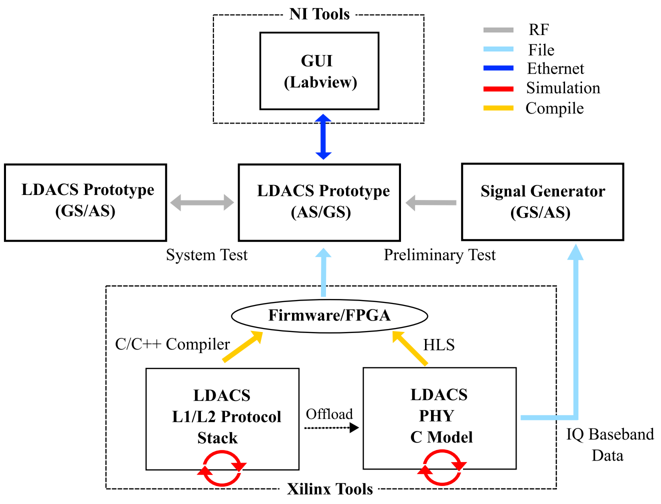


FIGURE 7. The development and debugging architecture. The firmware and FPGA are developed by Xilinx tools and the user interface is developed using a National Instruments (NI) tool.

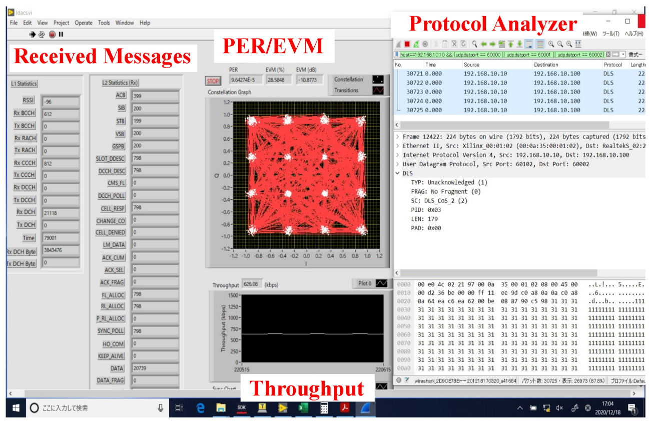


FIGURE 8. The user interface of the prototype. The GUI displays graphical representations of the number of received messages, PER, EVM, and throughput. The protocol analyzer provides detailed information on the transmitted/received LDACS messages.

IPv6 protocol stack was implemented using lightweight IP (lwIP) [37], which is an open-source IP protocol stack for embedded systems. The other R5 core is reserved for future extensions.

Additionally, the Modulation (Mod)/Demodulation (Demod) and Synchronization (Sync) functions, which require time-critical computations, are implemented in the PL for load balancing. Initially, we implemented these modules using C code in the L1 core. Later on, we identified the three modules with the highest computational requirements and offloaded them to the FPGA to enable real-time operation. Here, HLS was used to implement the FPGA to re-use the C code and to reduce development costs. Implementation details are described in Section IV-C.

Although the original software may have to be modified for the protocol stack to meet the specifications of other aeronautical systems, we can still reuse the hardware and driver software, including the intercore, application, and FPGA interfaces. This approach can significantly reduce the development costs associated with developing other aeronautical system prototypes.

Fig. 7 illustrates a flow diagram depicting the software development and debugging process. First, we developed the L1/L2 protocol stack in C source code. Then, we verified the functionality of the developed code with unit tests. After that, the compiled program was loaded and executed on the target CPUs. Then, the modules which required the most computational resources were identified. Finally, these modules, i.e., synchronization, modulation, and demodulation modules, were offloaded to the FPGA.

For rapid and cost-effective prototype development, we employed HLS to implement the FPGA. By utilizing HLS, we efficiently implemented the FPGA by reusing the C code that was previously used for the simulation. This was achieved by incorporating a few preprocessors within the C code, without resorting to HDL, which is typically challenging for development and debugging.

Thus, the L1/L2 protocol stack and offloaded C model can be debugged and simulated offline. Once debugging and simulation are finished, the L1/L2 firmware and the FPGA configuration file can be generated using the Xilinx tools.

Furthermore, the generation of IQ baseband test data was implemented in C code. By using the IQ baseband test data, we can generate the test RF signal using a Signal Generator (SG). With this function, we can perform preliminary tests efficiently using the SG, before conducting the overall system tests between the prototype AS and GS.

A measurement application was developed to evaluate the prototype. We used LabVIEW, a graphical programming language by National Instruments Corp., to develop the GUI. Fig. 8 shows an overview of the measurement application. Using the developed measurement application, we measured throughput, Error Vector Magnitude (EVM), Packet Error Rate (PER), and the number of transmitted and received messages. In addition, we used Wireshark [38], which is an open-source protocol analyzer, to evaluate packet transmissions/receptions on every protocol layer. Wireshark can handle new protocols, such as the LDACS, by describing the protocol definition using the script language Lua. Further, its packet visualizing/filtering functions facilitate efficient debugging.

C. IMPLEMENTATION

As described in Sections IV-A and IV-B, we used HLS to implement the FPGA. The HLS is an essential part of our software/hardware co-design. Fig. 9 shows the FPGA design flow.

First, a high-level behavioral model is described using C/C++ or System C. After the code is tested through unit tests, the high-level description is manually translated into a Register Transfer Level (RTL) description in the general FPGA design flow, whereas in the HLS design flow, this translation is done by the HLS tool automatically.

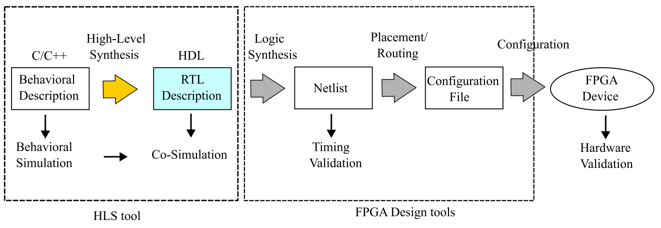


FIGURE 9. The FPGA design flow. The HLS tool generates RTL descriptions automatically from the high-level behavioral description written in C/C++ or System C language.

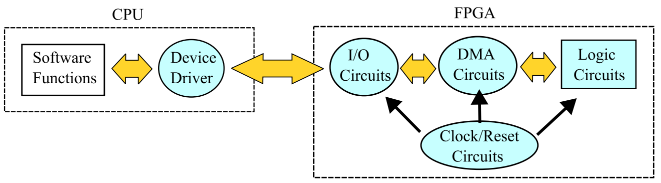


FIGURE 10. The interface between CPU and FPGA. The HLS tool generates not only the target logic circuits but also the device drivers and interface circuits, such as I/O, DMA, and clock and reset circuits automatically.

Second, after the RTL description is tested through RTL simulation in the general FPGA design flow, or through software/hardware co-simulation in the HLS design flow, it is translated into netlist format through logic synthesis. Here, the netlist describes the physical implementation of a design in terms of the specific components, such as FFs, LUTs BRAMs, DSP slices.

Third, once the timing validations are confirmed, the place-and-route tool is used to map the netlist to a specific FPGA device. This involves placing the components and routing the connections in accordance with the design requirements and constraints of the target device. The output of the place-and-route process is a configuration file that can be loaded on the target FPGA device.

Finally, the configuration file is loaded on the target FPGA, and the system behavior is tested using the hardware. In general, RTL descriptions are manually written using an HDL, which needs considerable development, and debugging effort. One advantage of using HLS is that FPGAs can be efficiently implemented in C code for software simulation by simply adding preprocessors to the code.

Fig. 10 shows the interface between the CPU and FPGA. In the general FPGA design flow, it is necessary to develop not only the target logic circuit but also the interface circuits, such as Input/Output (I/O), Direct Memory Access (DMA), Clock and Reset circuits, and the device driver software. Another advantage of the HLS design flow is that the HLS tool can produce these interface circuits and the device driver software automatically. Thus, the HLS design flow dramatically reduces the development cost of the FPGA.

Here we provide examples of preprocessors (also referred to as pragmas) that were used in our design. Additionally, we share a useful tip for using HLS and show the implementation results. The pragmas used here are specific to Xilinx FPGAs [39]. However, a similar approach can be used for other FPGAs such as Intel FPGAs.

The main difference between a CPU and an FPGA is that a CPU performs processing sequentially, whereas an FPGA can process data in parallel. Parallelization is an essential feature of FPGA design. Therefore, here, we introduce three pragmas that can be used to achieve parallelization:

- #pragma HLS PIPELINE
- #pragma HLS DATAFLOW
- #pragma HLS UNROLL

The 'HLS PIPELINE' pragma is used to optimize the implementation of a digital circuit by exploiting parallelism in the design. This pragma indicates that the circuit should be implemented as a pipeline. That is, the procedure is divided into small processing stages and the stages should be executed in parallel to increase the clock frequency, which in turn reduces the latency of the design and improves its overall performance. Generally, this pragma is used within a loop which differs from the 'DATAFLOW' pragma below.

The 'HLS DATAFLOW' pragma is also used to optimize the implementation of a digital circuit by exploiting flow-level parallelism in the design. Here, flow-level means a series of functions, statements, data processing, and so on. This pragma indicates that the computation should be performed as soon as the inputs are available, without waiting for the previous flow. This improves overall performance in terms of latency.

The 'HLS UNROLL' pragma is used to optimize the implementation of a loop by unrolling it. This pragma indicates that the loop should be replicated a specific number of times so that the loop iterations are executed in parallel. This can improve performance in terms of latency at the cost of increasing hardware resources. While the 'PIPELINE' pragma divides the circuit into small pieces of processing units and parallelizes them, the 'UNROLL' pragma generates the specified number of replicated circuits.

Although these three pragmas can be used to achieve parallelization, the HLS tool may not always produce the expected parallelized circuits, especially when the code includes memory accesses. This issue is commonly called the 'memory bandwidth bottleneck problem'. Here, we present an example of the memory bandwidth bottleneck problem that we encountered and explain how we solved it. This tip can be very useful for readers who are interested in using HLS.

During our initial implementation of the FPGA using HLS, we encountered a long latency issue with the Viterbi algorithm [35], [36], which was not acceptable for real-time signal processing. The Viterbi algorithm is a widely used decoding technique in digital communication systems, particularly for decoding convolutional codes. Fig. 11 shows a general flowchart of the Viterbi algorithm (from [36, Fig. 2]). It works by constructing a trellis diagram to represent all possible sequences of coded bits ('Initialize' in Fig. 11) and finding/storing the most likely sequence by calculating the

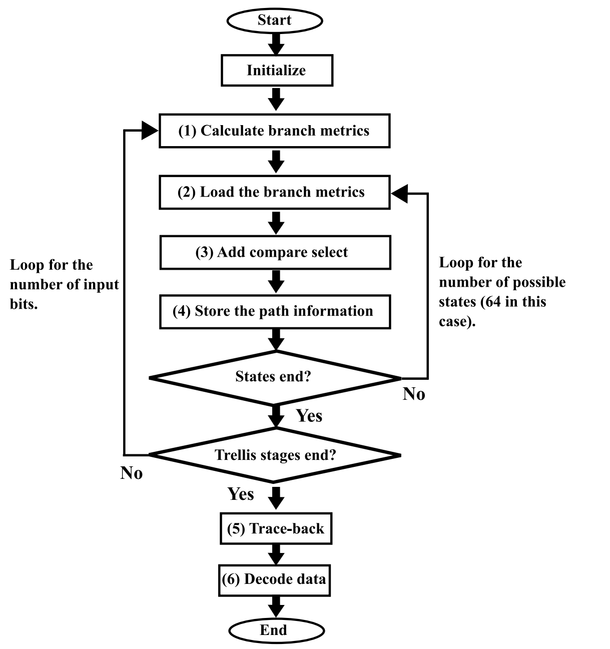

Start
Initialize
(1) Calculate branch metrics
(2) Load the branch metrics
(3) Add compare select
(4) Store the path information
States end?
No
Yes
Trellis stages end?
No
Yes
(5) Trace-back
(6) Decode data
End
Loop for the number of input bits.
Loop for the number of possible states (64 in this case).

FIGURE 11. A general flowchart of the Viterbi algorithm taken from [36]. The figure has been modified to explain Fig. 12.

```
#pragma HLS ARRAY_PARTITION dim=1 factor=32 type=cyclic variable=paths
// Path metric calculation
for (number of input bits)
  #pragma HLS UNROLL
  // calculate cur_met_0 and cur_met_1 from received symbols
  cur_met_0 = metric( symbols[0] );   ] (1) in Fig.11
  cur_met_1 = metric( symbols[1] );
  symbols += 2;                                          (2) (3) (4) in Fig.11

  // calculate path metric for each state and store to the path memory
  path_memory[0] = metric_calloc( acc_met[0] + cur_met_0, acc_met[0] + cur_met_1 );
  path_memory[1] = metric_calloc( acc_met[1] + cur_met_0, acc_met[1] + cur_met_1 );
  path_memory[2] = metric_calloc( acc_met[2] + cur_met_0, acc_met[2] + cur_met_1 );
  path_memory[3] = metric_calloc( acc_met[3] + cur_met_0, acc_met[3] + cur_met_1 );
  ...
  path_memory[60] = metric_calloc( acc_met[60] + cur_met_0, acc_met[60] + cur_met_1 );
  path_memory[61] = metric_calloc( acc_met[61] + cur_met_0, acc_met[61] + cur_met_1 );
  path_memory[62] = metric_calloc( acc_met[62] + cur_met_0, acc_met[62] + cur_met_1 );
  path_memory[63] = metric_calloc( acc_met[63] + cur_met_0, acc_met[63] + cur_met_1 );

  path_memory += 64;

end

// Trace back through the most likely pathTrace back
for (number of input bits)
  #pragma HLS UNROLL
  // trace back the path_memory and estimate the symbols  ← (5) (6) in Fig.11
end
```

FIGURE 12. Pseudocode of the Viterbi algorithm. Within the for loop, the algorithm computes the path metric for every possible state. The calculations for each state are independent of each other. (1) - (6) correspond to the steps of the flowchart in Fig. 11.

probabilities for each path (1-4 in Fig. 11). The algorithm then decodes the original information bits by tracing back through the most likely path (5-6 in Fig. 11).

Fig. 12 shows the pseudocode of the Viterbi algorithm that we implemented in our design. In the for loop, the algorithm calculates the path metric for each possible state. Since the calculation for each state is independent of the calculations for the other states, we expected the generated

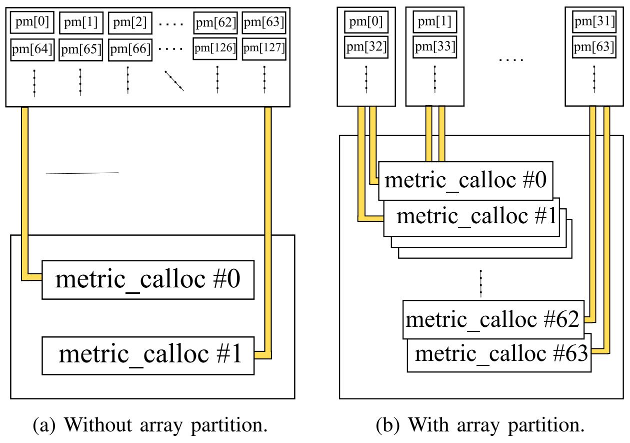


FIGURE 13. A graphical representation of the generated path memory (a) without and (b) with array partition. In each memory segment, the memory access is limited to two accesses at the same time.

circuits to process these calculations in parallel for each state simultaneously. However, the generated circuits performed simultaneous calculation for only two states rather than processing all states in parallel. This occurred because the access to the path memory caused a memory bandwidth bottleneck problem. The memory in our design had only two ports for access, which limited the memory accesses to two at a time. If the code has sequential memory access, the 'HLS ARRAY_PARTITION' pragma can be used to solve the memory bandwidth bottleneck issue.

- #pragma HLS ARRAY_PARTITION

The 'HLS ARRAY_PARTITION' pragma is a directive that indicates the array should be divided into smaller sub-arrays, called partitions, to improve memory access performance. Partitioning an array can lead to improved performance in terms of speed and power consumption as it allows for better utilization of memory resources and can also enable parallel processing. Using this pragma provides the HLS tool with information on how to divide the array for optimal performance and helps to specify the parallelism in the design.

In the Viterbi algorithm described in Fig. 12, the path memory is computed and stored in memory to be used for the traceback function later on. Fig. 13 shows images of the generated circuits with and without the 'HLS ARRAY_PARTITION' pragma. The figures illustrate that memory access is limited to only two times at the same time when the pragma is not used as in Fig. 13 (a), whereas memory access is allowed up to 64 times at the same time when the pragma is used as in Fig. 13 (b).

Table 1 presents a comparison of the latency and resource utilization with and without the 'HLS ARRAY_PARTITION' pragma. The results indicate that the latency was significantly reduced from 193 μs to 6.06 μs with only a 0.71% increase in LUT utilization. This example demonstrates the importance of being careful of potential memory bandwidth bottleneck problems when utilizing HLS in FPGA design.

TABLE 1. Latency and resource usage with and without array partition. The numbers indicate the actual count values. The time, given in 100 MHz clock cycles, and the percentages of the total resources are shown in the parentheses.

	Cycles (us)	LUT (%)	FF (%)	BRAM (%)	DSP (%)
Without PARTITION	19300 (193)	8239 (3.01)	3792 (0.69)	3 (0.33)	0 (0.00)
With PARTITION	606 (6.06)	10199 (3.72)	3636 (0.66)	3 (0.33)	0 (0.00)

TABLE 2. Latency of demodulation module. The representative SDUs for the FL are BC1/BC3, BC2, and nine SDUs (64-QAM, 3/4 coding rate). The representative SDUs for the RL are RA, DC, and two SDUs (64-QAM, 3/4 coding rate). The frame length represents the duration of each SDU.

SDU TYPE	Cycles	Latency (ms)	Frame Length (ms)
BC1/BC3	33908	0.34	1.8
BC2	56802	0.57	3.12
FL 9 SDU 64QAM-3/4	588345	5.88	19.44
RA	59448	0.59	0.84
DC	22535	0.23	0.72
RL 2 SDU 64QAM-3/4	21851	0.22	0.72

D. RESULTS

To confirm that the implemented circuits meet the real-time requirements, we measured the overall decoding latencies of various types of LDACS PHY-SDUs through software/hardware co-simulation after the synthesis phase. We only measured the latency for de-modulation because it generally has a higher computational complexity than modulation. If the de-modulation is completed within the frame length, then real-time processing is possible.

Table 2 presents a comparison of the latency of the de-modulation module. From the table, we can see that the latencies for all types of SDUs for both the FL and the RL were much smaller than the frame length. Note that the latency should be shorter than the frame length not only for the de-modulation process but also for the processing of the L2/L3 protocols. While the processing latency of the L2/L3 protocols contributes, the majority of latency is attributed to the physical layer de-modulation. The co-simulation outcomes strongly suggest the feasibility of real-time decoding. In Section V, we validate the end-to-end IPv6 real-time communication through a laboratory test.

Additionally, to confirm whether the implemented circuits fulfilled the resource requirements, we analyzed the estimated resource utilization after synthesis and the actual resource utilization after implementation. Table 3 shows the resource utilization estimates of each module after synthesis. The table clearly illustrates that the de-modulation module utilized the highest amount of FPGA resources, aligning with our expectations. Further, we can see that the total utilization of LUT became 103.2% after synthesis.

TABLE 3. Resource utilization estimated after synthesis for each module. The numbers indicate the actual count values, and the percentages of the total resources are shown in the parentheses.

	LUT (%)	FF (%)	BRAM (%)	DSP (%)
Sync	5147 (1.9)	3977 (0.7)	615 (33.7)	10 (0.4)
Mod	53066 (19.4)	24733 (4.5)	360 (19.7)	45 (1.8)
Demod	224453 (81.9)	71584 (13.1)	224 (12.3)	119 (4.7)

TABLE 4. Total resource utilization after implementation. The numbers indicate the actual count values, and the percentages of the total resources are shown in the parentheses.

	LUT (%)	FF (%)	BRAM (%)	DSP (%)
Total	168076 (61.3)	102753 (18.7)	621.5 (68.2)	351 (13.9)

TABLE 5. Comparison of three different software/hardware co-design approaches for LDACS prototyping.

	Proposed (This paper)	N. Agrawal et.al [32]	S. Kurz et.al [33]
Implemented Functions	PHY L2/L3	PHY	PHY L2/L3
Behavioral Description	C	Simulink	System C
RTL Description	Automatic (HLS)	Automatic (HDL coder)	Manual (VHDL)
Real-time Factor (FL 64QAM-3/4)	3.31	–	9.95
Resource Utilization (Total LUT in %)	61.3	–	63.0

Table 4 shows the actual resource utilization after implementation. Even though the total utilization of LUT was over 100% after synthesis, it was optimized during the place-and-route process and the actual total utilization became 61.3%, as shown in the table. We can also see that the actual utilization of FF, BRAM, and DSP were 18.7%, 68.2%, and 13.9%, respectively. This implies that there is still enough room to implement additional functions that require computational resources.

E. DISCUSSIONS

Here, we discuss the differences between our approach and those of others [32], [33]. The three prototypes utilized Xilinx MPSoC as their signal processing module, but they adopted different software/hardware co-design approaches. The differences are summarized in Table 5.

First, the purpose of our prototype and that of [33] is to realize real-time End-to-End (E2E) communications. Conversely, the prototype described in reference [32] primarily concentrates on assessing the interference between LDACS and DME. Therefore, our prototype and that of [33] implement all protocol stacks including PHY, L2 and L3, whereas the prototype in [32] implements only PHY.

Second, our prototype and that of [33] used conventional C style programming language as behavioral description, whereas the prototype in [32] utilized graphical model-based programming language (i.e., Simulink by Mathworks). Simulink allows users to create models using a block diagram approach, making it easier to visualize (cf. [32, Figs. 5, 7 and 9]). The visual nature of the graphical-based modeling simplifies programming, therefore it seems to be an attractive candidate for rapid prototyping if the model meets real-time and resource requirements. In contradiction, building too intricate models (e.g., RS decoder or Viterbi decoder) in graphical-based modeling can sometimes lead to complicated diagrams that are difficult to manage.

Third, our approach and that of [32] generated the RTL description automatically using HLS and HDL coder respectively, whereas the approach in [33] developed the RTL description manually by VHDL. As we described in Section IV-C, and also as noted in [33], manual development of an RTL description by HDL has disadvantages such as a higher implementation and testing effort and inferior maintainability of the system.

Next, we compare latency and resource utilization. A completely equitable comparison among different systems is difficult to achieve due to variations in implementation completeness and the potential for further optimization in certain cases. Especially, we couldn't see any description concerning RS codes and convolutional codes in [32], therefore it is unclear if they implemented them. Further, throughput evaluations have not been conducted in [32], therefore it is also unclear whether their implementation meets real-time requirements. Please note again, their purpose is not to realize real-time communications, but to evaluate interference between LDACS and DME. On the other hand, an interoperability test for the FL has been conducted between our prototype and that of [33] (cf. Section IV-E), so these two prototypes have compatibility in their implementation. Therefore, we compare latency and resource utilization only for ours and that of [33].

The real-time factor can be calculated either by dividing the achieved latency by the required latency or by dividing the achieved data rate by the required data rate. Values greater than one indicate that the real-time requirements are satisfied, and a higher value indicates better performance. From Table 2, we can see that the real-time factor of our prototype for FL 64QAM-3/4 is 3.31 (= 19.44/5.88). On the other hand, that of [33] is 9.95 (cf. [33, Table 4]). This is likely attributed to their manual optimization of the RTL description, which, although demanding in terms of development effort, results in efficient performance. However, note that a value greater than one is sufficient to meet the real-time requirement.

Finally, resource utilization of the LUT of the overall system including data processing, synchronization, and more, is 61.3% for our system (cf. Table 4), and 63.0% for [33] (cf. [33, Sec. 5.A]). We can see that although the RTL description was generated automatically in our approach, the resource utilization is almost the same as that of a manually optimized RTL description of [33].

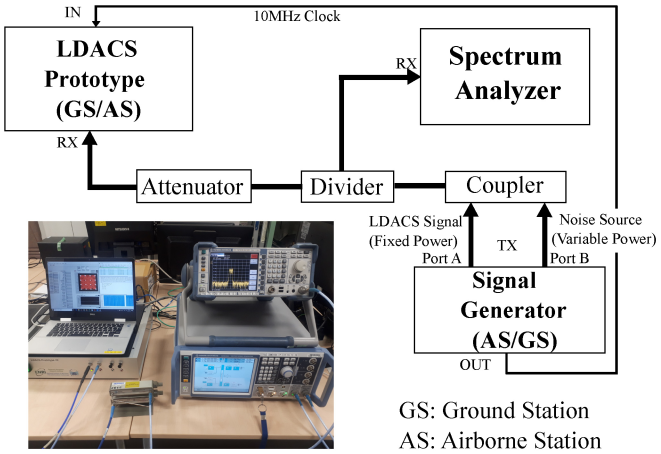


FIGURE 14. Test setup for the preliminary test. FL and RL signals from the SG are received by the prototype AS and GS, respectively. The SNR is changed by adjusting the transmit power of port B of the SG.

V. VALIDATION TESTS

In the previous section, we presented a detailed design of our LDACS prototype. In this section, we validate that the prototype operates in real-time and according to the specification [11]. The outcomes of our validation tests were provided as inputs in the standardization process at the ICAO.

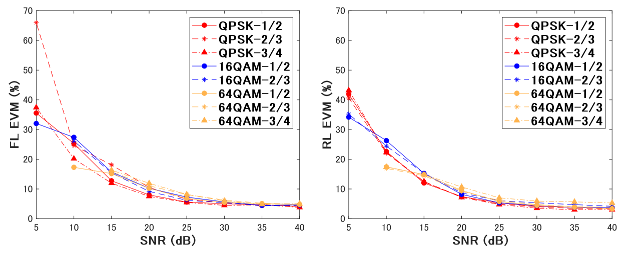
A. PRELIMINARY TEST

Prior to conducting counter tests between the prototype AS and GS, we conducted preliminary tests in a laboratory using the SG to evaluate their basic performance. We confirmed through this test that our prototype was correctly implemented according to the LDACS specification and operated in real-time.

Fig. 14 displays the test setup used for the preliminary test. In this setup, the LDACS signal was generated at port A of the SG, and an Additive White Gaussian Noise (AWGN) signal was generated at port B. The bandwidth of the AWGN was 75 MHz, which was significantly larger than the LDACS signal bandwidth of 500 kHz.

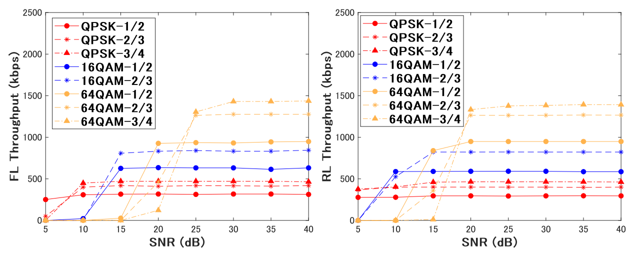
The transmit power of port A was kept constant at −26 dBm to intentionally alter the SNR by adjusting the transmit power of port B. The relationships between the SNR and EVM, between the SNR and PER, and between the SNR and throughputs were then evaluated.

During this evaluation, the prototype AS/GS and the SG shared a 10 MHz reference clock, to assess the hardware's



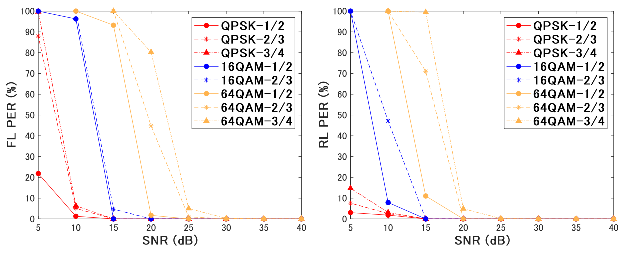
(a) EVM for FL. (b) EVM for RL.

FIGURE 15. EVM vs. SNR on the FL and RL. The vertical axis represents the EVM in percentage, which indicates the deviation of the received signals from the ideal ones.



(a) Throughputs on the FL. (b) Throughputs on the RL.

FIGURE 17. Throughputs vs. SNR on the FL and RL. The vertical axis represents throughputs in kbps.



(a) PER for FL. (b) PER for RL.

FIGURE 16. PER vs. SNR on the FL and RL. The vertical axis represents the PER in percentage.

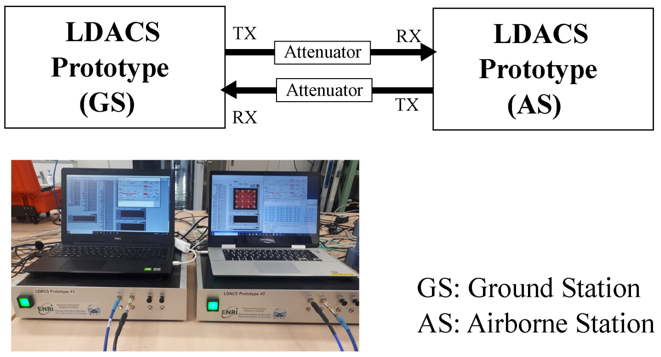


FIGURE 18. The configuration for the overall system tests between the AS and GS prototypes. The independent 10 MHz reference clock is provided from each clock module.

performance limit under ideal conditions. In other words, we ensured that time and frequency synchronization between the prototype AS/GS and the SG.

Fig. 15 shows the evaluation results of the EVM on the (a) FL and (b) RL, respectively. The EVM measures the deviation rate from the ideal signal, thus, a smaller value represents better performance. The figure indicates that the EVM reduces as the SNR increases. However, the EVM decrease saturates at approximately 5%, even when the SNR exceeds 30 dB. In this evaluation, we expected the EVM to be less than 1% when the SNR was sufficiently high because the prototypes and the SG were connected directly and shared the reference clock. The results show that the hardware's performance limit was approximately 5% of the EVM.

Fig. 16 shows the PER vs. SNR on the (a) FL and (b) RL, respectively. The results demonstrate that the PER approaches zero as the SNR increases. These results show that even though the hardware's performance limit was approximately 5% of the EVM, it is enough for decoding signals correctly when the SNR is sufficiently high.

Fig. 17 shows the throughputs vs. SNR on the (a) FL and (b) RL, respectively. The results demonstrate that our prototype can achieve the specified LDACS throughputs as per its specification [11] when the SNR is adequately high. Based on these findings, it is evident that the LDACS FL/RL physical layer frame formats are appropriately implemented according to the LDACS specification, confirming the prototype's capability to decode LDACS signals in real-time.

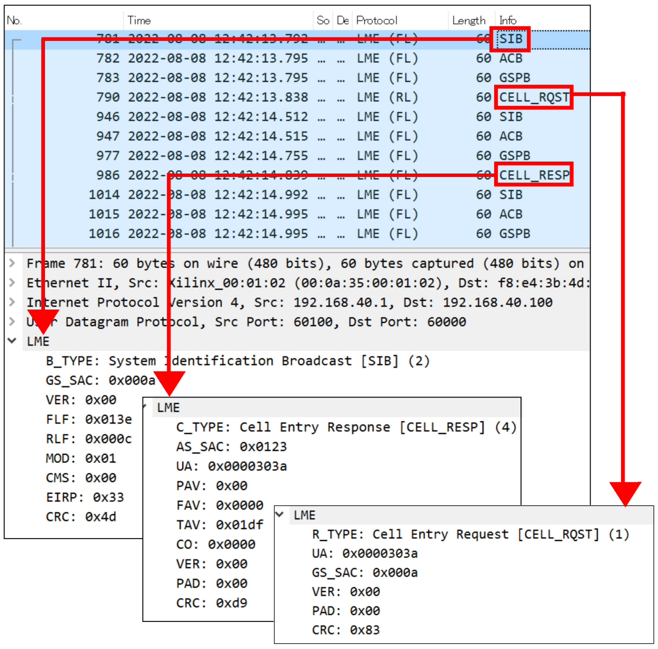
B. CELL ENTRY TEST

In Section V-A, we analyzed the performance limitations of the selected hardware. Here, we evaluate our LDACS prototype as an overall system test from the protocol behavior perspective.

Fig. 18 shows the configuration used for the overall system test. The TX port of the GS was directly connected to the RX port of the AS, and the TX port of the AS was directly connected to the RX port of the GS using RF cables. The transmit power of the TX port was set to -10 dBm, and 50 dB attenuators were placed between the TX and RX ports. During the tests, an independent reference clock (10 MHz) from each clock module was provided to evaluate the timing and frequency correction functions.

Here, we briefly summarize the cell entry process. First, the AS synchronizes its time and frequency using the synchronization symbols of the BCCH transmitted from the GS. After synchronization, the AS decodes the SIB message to obtain the system parameters necessary for air-to-ground communication. The AS then sends a cell entry request message (i.e., CELL_RQST) via the RACH, including a UA which serves as an aircraft identifier.

Once the GS receives the request message for cell entry and verifies that the UA exists, it responds by sending a cell entry response message (i.e., CELL_RESP). This response message contains information, such as FAV, TAV, and PAV that is required to adjust the frequency, time, and power differences between the GS and the AS.



```
No.    Time                         So De Protocol   Length Info
 781 2022-08-08 12:42:13.792 …  …  LME (FL)       60 SIB
 782 2022-08-08 12:42:13.795 …  …  LME (FL)       60 ACB
 783 2022-08-08 12:42:13.795 …  …  LME (FL)       60 GSPB
 790 2022-08-08 12:42:13.838 …  …  LME (RL)       60 CELL_RQST
 946 2022-08-08 12:42:14.512 …  …  LME (FL)       60 SIB
 947 2022-08-08 12:42:14.515 …  …  LME (FL)       60 ACB
 977 2022-08-08 12:42:14.755 …  …  LME (FL)       60 GSPB
 986 2022-08-08 12:42:14.839 …  …  LME (FL)       60 CELL_RESP
1014 2022-08-08 12:42:14.992 …  …  LME (FL)       60 SIB
1015 2022-08-08 12:42:14.995 …  …  LME (FL)       60 ACB
1016 2022-08-08 12:42:14.995 …  …  LME (FL)       60 GSPB

> Frame 781: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on
> Ethernet II, Src: Xilinx_00:01:02 (00:0a:35:00:01:02), Dst: f8:e4:3b:4d:
> Internet Protocol Version 4, Src: 192.168.40.1, Dst: 192.168.40.100
> User Datagram Protocol, Src Port: 60100, Dst Port: 60000
v LME
    B_TYPE: System Identification Broadcast [SIB] (2)
    GS_SAC: 0x000a
    VER: 0x00
    FLF: 0x013e
    RLF: 0x000c
    MOD: 0x01
    CMS: 0x00
    EIRP: 0x33
    CRC: 0x4d
```

```
v LME
    C_TYPE: Cell Entry Response [CELL_RESP] (4)
    AS_SAC: 0x0123
    UA: 0x0000303a
    PAV: 0x00
    FAV: 0x0000
    TAV: 0x01df
    CO: 0x0000
    VER: 0x00
    PAD: 0x00
    CRC: 0xd9
```

```
v LME
    R_TYPE: Cell Entry Request [CELL_RQST] (1)
    UA: 0x0000303a
    GS_SAC: 0x000a
    VER: 0x00
    PAD: 0x00
    CRC: 0x83
```

FIGURE 19. Cell entry logs captured by Wireshark. The detailed parameters of SIB, CELL_RQST, and CELL_RESP are shown.

After the AS receives the cell entry response, it adjusts the frequency, timing, and power offsets and transitions to the connected state. Once in the connected state, the GS and AS can communicate upper-layer user data.

Fig. 19 shows the logs obtained by Wireshark during the cell entry process. The figure illustrates three messages related to the cell entry process, which are the SIB on the FL, CELL_RQST on the RL, and CELL_RESP on the FL. The logs showed that the prototype GS and AS successfully completed cell entry.

Note that the cell entry process must be secured. However, at the time of these experiments, our prototype did not implement security functions. The detailed security functions including a secured cell entry process were described and evaluated by simulation in [19]. Further, the security functions were also evaluated through an emulation using our prototype in [22], as summarized in Section V-D. The actual security implementation in our prototype is left for future work.

C. END-TO-END TEST

After the successful cell entry process, E2E IPv6 data communication between the GS and AS sides becomes possible. Here, the results of IPv6 E2E tests over LDACS are shown. Please note that this work was conducted in a joint effort between ENRI and the DLR. Fig. 20 shows the test setup for the E2E tests. The setup is the same as the cell entry test in the previous subsection except that additional application PCs are connected via a network hub on each side.

We used a synthetic application data traffic generator developed by the DLR to emulate a realistic aeronautical data traffic pattern. Therefore, few large and many small packets

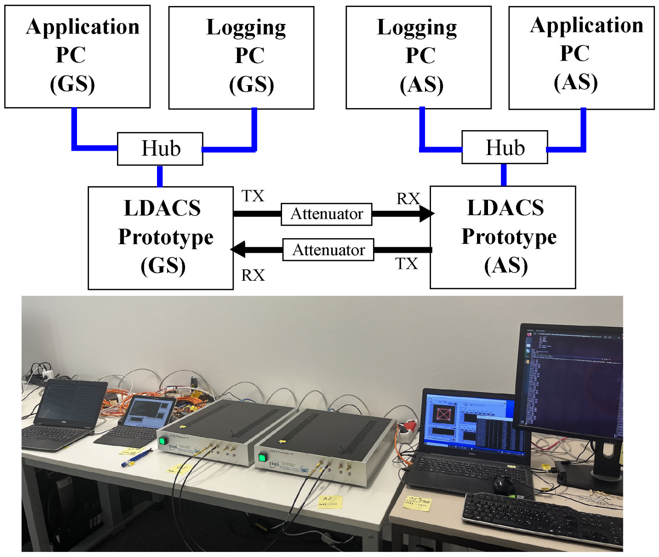


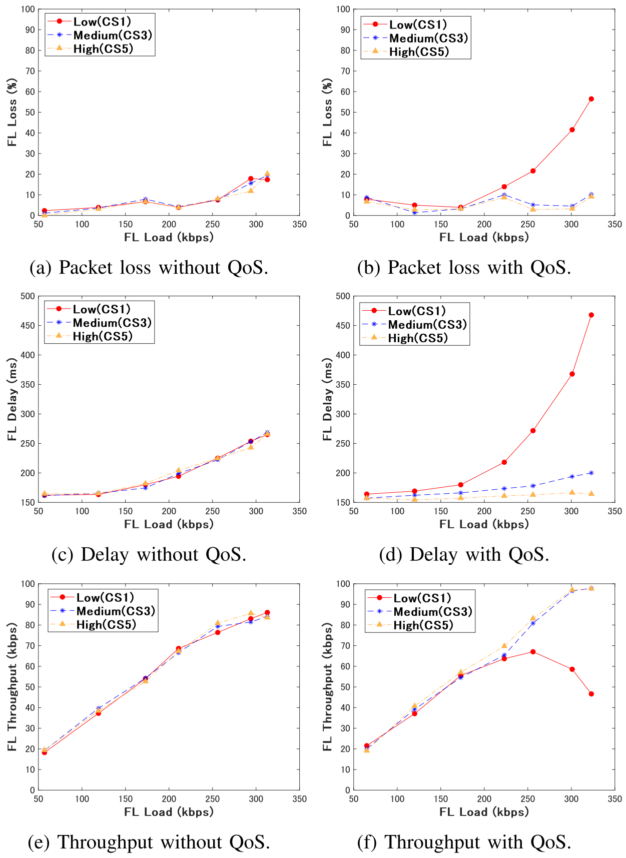
FIGURE 20. The test setup for the E2E tests. The DLR application PCs are connected to the ENRI prototype via a network hub on each side.

were generated as ATS and Aeronautical Operational Control (AOC) application equivalents following a typical aeronautical data traffic pattern [40]. These were transmitted over the LDACS prototypes at various throughput rates, and utilizing different coding and modulation rates. A detailed description of the generator and evaluation is shown in [41].

Layer 3 of the LDACS prototype has the capability to manage Quality of Service (QoS) using IP Differentiated Services (DiffServ). At the end-hosts, each application is assigned a unique class of service, which is communicated in the network through the IPv6 Differentiated Services Code Point (DSCP). For example, a DSCP value of Class Selector 1 (CS1) is used to indicate the lowest priority level, with higher class selectors indicating higher priority levels. The LDACS radio then associates the IPv6 DSCP with the appropriate LDACS class of service.

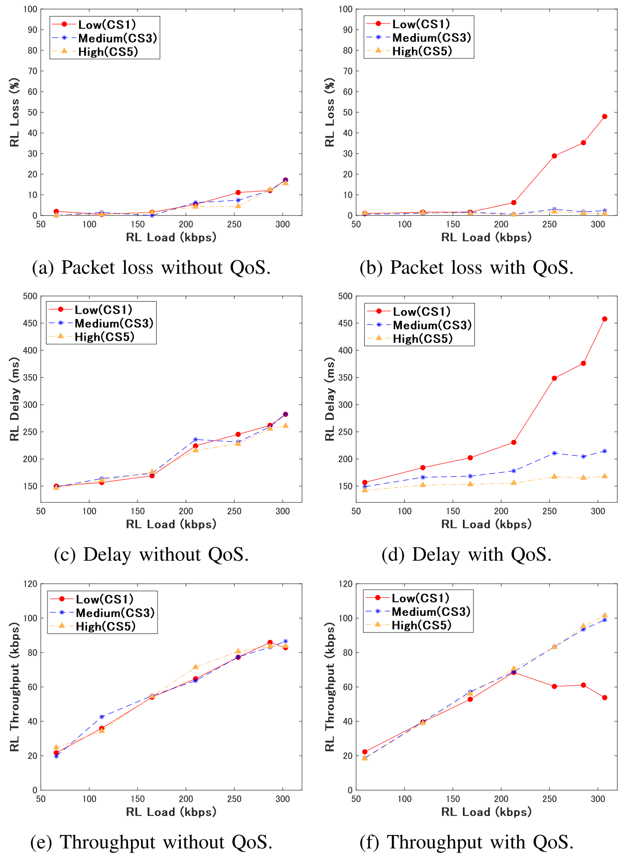
In this experiment, three classes of service were created using DLR synthetic application data traffic generators: high (CS5), medium (CS3), and low (CS1). The packet sizes for these classes were 1380 bytes, 1400 bytes, and 1420 bytes, respectively. To demonstrate the QoS functionality of our prototypes, we gradually increased the traffic load from 60 kbps to 320 kbps, both with and without QoS. We measured the packet loss rate, delay, and throughput on both the FL and RL.

Fig. 21 shows the results for the FL. Fig. 21 (a) and Fig. 21 (b) display the packet loss rates without and with QoS, respectively. These figures indicate that, when the QoS function was disabled, the packet loss for each priority level increased proportionally as the data load increased. However, when the QoS function was enabled, the packet loss for high and middle-priority levels increased less than that for the low-priority level, even as the data load increased. This is the expected behavior of the protocol.



(a) Packet loss without QoS. (b) Packet loss with QoS. (c) Delay without QoS. (d) Delay with QoS. (e) Throughput without QoS. (f) Throughput with QoS.

FIGURE 21. FL QoS test results. The horizontal axis represents the FL load in kbps. Subplots (a) and (b) compare packet loss, (c) and (d) compare delay, and (e) and (f) compare throughput, both without and with the QoS function.



(a) Packet loss without QoS. (b) Packet loss with QoS. (c) Delay without QoS. (d) Delay with QoS. (e) Throughput without QoS. (f) Throughput with QoS.

FIGURE 22. RL QoS test results. The horizontal axis represents the RL load in kbps. Subplots (a) and (b) compare packet loss, (c) and (d) compare delay, and (e) and (f) compare throughput, both without and with the QoS function.

Fig. 21 (c) and Fig. 21 (d) illustrate the delay measurements without and with QoS, respectively. Again, the results show that, when the QoS function was disabled, the delay for each priority level increased proportionally as the data load increased. However, when the QoS function was enabled, the delay for high and middle-priority levels increased less than that for the low-priority level, even as the data load increased. This is also an expected behavior of the protocol.

As a result, we can observe from Fig. 21 (e) and Fig. 21 (f) that the throughputs for all priority levels reached saturation when the QoS function was disabled. However, when the QoS function was enabled, the throughput for high and middle-priority levels did not saturate, at the expense of the throughput for the low-priority level, as intended.

Fig. 22 presents the RL results. From these figures, we can observe the same trends as seen in the FL results. These results clearly show that the QoS handling in our prototype works successfully on both the FL and RL.

D. SECURITY TEST

Since the LDACS prototype AS and GS did not include security features, these were emulated by two separate Linux laptops utilizing the same setup as the E2E test in Fig. 20. The LDACS cybersecurity architecture foresees the MAC, LME, and SNP protocol layer to be extended by substantial security features, such as the Mutual Authentication and Key Establishment (MAKE) protocol and user and control data protection measures [18], [19], [22]. Since control channel protection measures require changes in the MAC and LME layer, keys were only negotiated for those security measures without adapting the actual BCCH, CCCH, or DCCH content. However, the secure cell entry and MAKE protocol, including actual user-data protection measures were implemented using the Python 3 *cryptography* and *PyCryptodome* libraries.

```
// Set GS keys at 0.2468328 s
kAS_GS: b'\x93 \x00^\xd1\x85Y\xfetO\xe3\x1b\x8b-CB'

// Set AS keys at 0.3229198 s
kAS_GS: b'\x93 \x00^\xd1\x85Y\xfetO\xe3\x1b\x8b-CB'

// AS AEAD encrypts and sends the packet
m: b'The quick brown ...'
c: b'"?N\x07\xe4\xeb\xc2\x16\xec\xfe\xf6\xf3[\x17\x81'

// GS receives the packet and AEAD decrypts
c: b'"?N\x07\xe4\xeb\xc2\x16\xec\xfe\xf6\xf3[\x17\x81'
m: b'The quick brown ...
```

Listing 1. Shared Keys After MAKE Used to Secure LDACS User-Data.

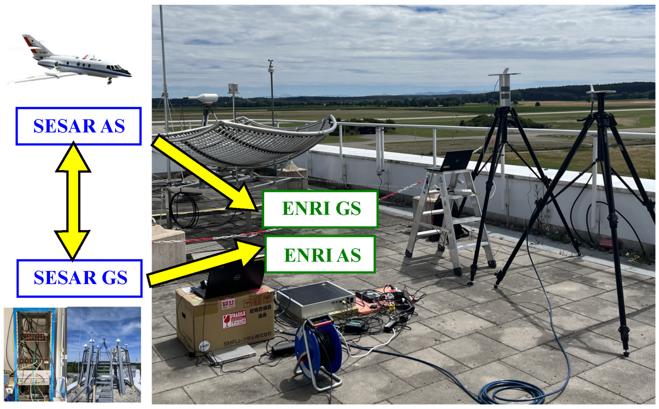
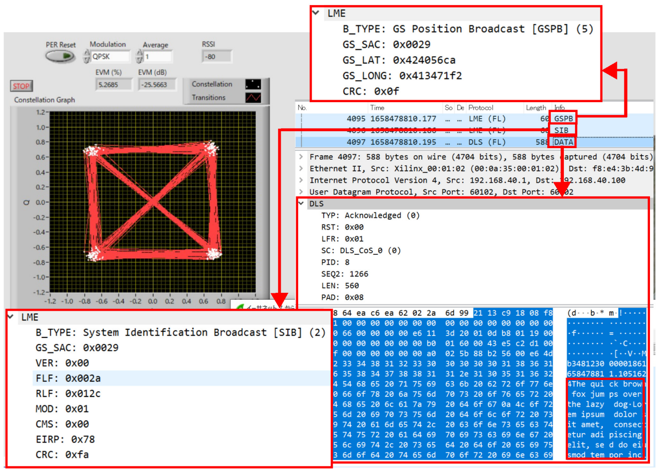


FIGURE 23. Interoperability test setup. The ENRI prototypes work in receiver-only mode. FL/RL signals from the SESAR GS/AS were received by the ENRI AS/GS.



```
LME
    B_TYPE: GS Position Broadcast [GSPB] (5)
    GS_SAC: 0x0029
    GS_LAT: 0x424056ca
    GS_LONG: 0x413471f2
    CRC: 0x0f
```

```
No.              Time   So De Protocol  Length Info
     4095 1658478810.177 … … LME (FL)      60 GSPB
     4096 1658478810.186 … … LME (FL)      60 SIB
     4097 1658478810.195 … … DLS (FL)     588 DATA
> Frame 4097: 588 bytes on wire (4704 bits), 588 bytes captured (4704 bits)
> Ethernet II, Src: Xilinx_00:01:02 (00:0a:35:00:01:02), Dst: f8:e4:3b:4d:9
> Internet Protocol Version 4, Src: 192.168.40.1, Dst: 192.168.40.100
> User Datagram Protocol, Src Port: 60102, Dst Port: 60102
DLS
    TYP: Acknowledged (0)
    RST: 0x00
    LFR: 0x01
    SC: DLS_CoS_0 (0)
    PID: 8
    SEQ2: 1266
    LEN: 560
    PAD: 0x08
```

```
8 64 ea c6 ea 62 02 2a  6d 99 21 13 c9 18 08 f8   (d···b·* m·!·····
1 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ········ ········
0 66 00 00 00 00 e6 11  3d 20 01 0d b8 01 19 00   ·f······ = ······
0 00 00 00 00 00 00 b0  01 60 00 43 e5 c2 d1 00   ········ ·`·C····
f 00 00 00 00 00 00 a0  02 5b 88 b2 56 00 e6 4d   ········ ·[··V··M
2 33 34 38 31 32 33 30  30 30 30 30 31 38 36 31   b3481230 00001861
6 35 38 34 37 38 38 31  31 2e 31 30 35 31 36 32   65847881 1.105162
4 54 68 65 20 71 75 69  63 6b 20 62 72 6f 77 6e   4The qui ck brow
0 66 6f 78 20 6a 75 6d  70 73 20 6f 76 65 72 20    fox jum ps over
4 68 65 20 6c 61 7a 79  20 64 6f 67 0a 4c 6f 72   the lazy  dog·Lo
5 6d 20 69 70 73 75 6d  20 64 6f 6c 6f 72 20 73   em ipsum  dolor
9 74 20 61 6d 65 74 2c  20 63 6f 6e 73 65 63 74   it amet,  consec
5 74 75 72 20 61 64 69  70 69 73 63 69 6e 67 20   etur adi piscing
5 6c 69 74 2c 20 73 65  64 20 64 6f 20 65 69 75   elit, se d do ei
3 6d 6f 64 20 74 65 6d  70 6f 72 20 69 6e 63 69   smod tem por inc
```

```
LME
    B_TYPE: System Identification Broadcast [SIB] (2)
    GS_SAC: 0x0029
    VER: 0x00
    FLF: 0x002a
    RLF: 0x012c
    MOD: 0x01
    CMS: 0x00
    EIRP: 0x78
    CRC: 0xfa
```

FIGURE 24. Decoded FL messages from the interoperability test. The detailed parameters of SIB, GSPB message, and text messages of the FL DATA from the SESAR GS are shown.

Listing 1 shows that the AS and GS derive the same keys after cell entry and MAKE and utilize this key for user-data protection by encrypting the message "The quick brown fox$\cdots$" using Authenticated Encryption with Associated Data (AEAD).

A detailed description and results of our security tests are given in [22].

E. INTEROPERABILITY TEST

Interoperability testing is important for eliminating ambiguity in standardization documents. Even with well-defined standards, there may be room for variations in interpretation resulting in implementation differences by various vendors or organizations. These ambiguities can be identified and resolved through interoperability testing, ensuring that all involved parties have a consistent understanding of the requirements and specifications. This helps reduce confusion and errors and ensures that communications systems can work seamlessly and efficiently with each other.

In the context of the SESAR project PJ.33-W3-02 FALCO, a flight campaign was conducted in July 2022, in collaboration among the DLR, Frequentis, Honeywell, and Airtel to assess the performance of LDACS radio prototypes. Two GSs were installed near the Oberpfaffenhofen airfield and one AS was installed in a Dassault Falcon 20 aircraft to communicate with the GSs during the flight. The objectives of the campaign were to demonstrate the feasibility of IPv6 traffic over LDACS and seamless handovers between the two GSs. A detailed description of the flight campaign is published in [12].

Fig. 23 shows the test setup for the interoperability test. Throughout the flight campaign, our ENRI prototypes functioned solely as receivers to avoid emitting radio waves, as they were not certified yet via necessary licenses for TX. The LDACS FL signals from the SESAR GS were received and decoded by the ENRI AS in real-time, whereas the RL signals from the SESAR AS were captured and stored by the ENRI GS for later offline analysis for the LDACS-WAM feasibility study [16]. Only FL signals were decoded in real-time because FL signals can be synchronized using the synchronization symbols in the BCCH. By contrast, synchronizing RL signals without conducting the cell entry is difficult.

Fig. 24 shows the decoded FL messages from the SESAR GS in real-time. Here, we see three messages in the figure, i.e., SIB, GSPB, and FL DATA. The GS_LAT and GS_LONG of the GSPB message showed 0x424056CA (48.08475) and 0x413471F2 (11.27781) in IEEE 754-2008 binary 32 format, which indicated the actual position where the GS was installed.

The "MOD" and "CMS" fields in the SIB message showed that the ACM mode was 1, which indicated cell-specific coding and modulation scheme, and that CMS was 0, which indicated a QPSK-1/2 coding scheme. Therefore, the actual payload, "DATA", was decoded automatically using QPSK-1/2 and the clear constellations of the QPSK were shown in the figure.

In addition, we could see a text message in DATA which displayed 'The quick brown fox jumps over the lazy dog$\cdots$'. The message was the same as transmitted by the ground applications. Note that the text message was in plain text since both the SESAR and ENRI prototypes did not implement security functions at this time.

The results showed that the ENRI AS could decode the FL messages from the SESAR GS. The results indicated that both SESAR and ENRI prototypes were correctly implemented according to the LDACS specification [11]. Some findings from this interoperability testing regarding the ambiguity of the specification, such as a definition of a Cyclic Redundancy Checksum (CRC) calculation and an interpretation of coding unit were feed-backed to the standardization process at the ICAO.

F. LIMITATIONS AND FUTURE WORKS

Here, we outline the current limitations of the ENRI prototype, along with our plans for future development. First, our LDACS prototype does not include security and digital voice features. However, these functions are essential components of future aeronautical communications systems. Therefore, we plan to implement, and evaluate these features in the near future.

Second, we are currently developing the RF front end of the ENRI prototype. In order to demonstrate its effectiveness in a real-world environment, further improvements to the RF characteristics are necessary. Specifically, we plan to replace the current AD/DA module (AD9361) with an ADRV9002, a new version of the SDR module manufactured by Analog Devices Inc, which features superior noise figure, phase noise, and dynamic range compared to the AD9361. After developing the RF front end, we will obtain a radio license in Japan and conduct field experiments.

Third, in our experiments, we used fixed resource allocations. However, how a GS manages radio resources in a multiple-AS environment in response to resource requests is an important functionality, and its implementation depends on vendors. Efficient resource management strategies will need to be studied and implemented in future work.

Finally, in our interoperability test, we only received FL signals from the SESAR GS. The analysis of recorded and stored RL signals is left for future work. Furthermore, we will focus on achieving real-time bi-directional interoperability between the ENRI and other prototypes, including control channel and data channel communication with and without security functions.

VI. CONCLUSION

In this paper, we presented our LDACS prototyping and international validation activities. First, we showed our prototype which was developed using a software/hardware co-design approach that incorporated an SDR and HLS. This approach facilitated rapid and cost-effective prototype development. We provided an example of a memory bandwidth bottleneck problem and demonstrated its resolution through array partitioning. The simulation and implementation results confirmed that our design met both the real-time operation requirements and resource requirements. Second, preliminary testing showed that our prototype achieved the maximum throughput, as specified by the LDACS specification. Moreover, Wireshark logs obtained during the cell entry process substantiated that the prototype successfully completed the cell entry process by exchanging necessary LDACS messages, such as SIB, CELL_RQST, and CELL_RESP. Third, the end-to-end IPv6 connectivity and security tests confirmed the proficiency of the QoS and security functions of LDACS. Finally, during interoperability testing between our Japanese and European prototypes, real-time decoding of messages such as, SIB, GSPB, and FL DATA highlighted the soundness and clarity of the LDACS specification.

ACRONYMS

A/G	Air/Ground
ACM	Adaptive Coding and Modulation
AEAD	Authenticated Encryption with Associated Data
AOC	Aeronautical Operational Control
APNT	Alternative Positioning Navigation and Timing
AS	Airborne Station
ATM	Air Traffic Management
ATS	Air Traffic Services
AWGN	Additive White Gaussian Noise
B2	Baseline 2
BC	Broadcast Control
BCCH	Broadcast Control CHannel
BRAM	Block RAM
CC	Common Control
CCCH	Common Control CHannel
CMS	Coding and Modulation Scheme
CPU	Central Processing Unit
CRC	Cyclic Redundancy Checksum
DC	Dedicated Control
DCCH	Dedicated Control CHannel
DCH	Data CHannel
Demod	Demodulation
DLR	German Aerospace Center
DLS	Data Link Service
DMA	Direct Memory Access
DME	Distance Measuring Equipment
DSCP	Differentiated Services Code Point
DSP	Digital Signal Processor
E2E	End-to-End
ENRI	Electronic Navigation Research Institute
EUROCAE	European Organisation for Civil Aviation Equipment
EVM	Error Vector Magnitude
FAV	Frequency Adaptation Value
FEC	Forward Error Correction
FBMC	Filter Bank Multi-Carrier
FF	Flip-Flop
FL	Forward Link
FPGA	Field Programmable Gate Array
GBAS	Ground Based Augmentation System
GS	Ground Station
GSPB	GS Position Broadcast
GUI	Graphical User Interface
HDL	Hardware Description Language
HLS	High Level Synthesis
I/O	Input/Output
IATA	International Air Transport Association
ICAO	International Civil Aviation Organization
I-CNS	Integrated-Communications, Navigation and Surveillance
IPS	Internet Protocol Suite
IPv6	Internet Protocol version 6
JTIDS	Joint Tactical Information Distribution System

LDACS	L-band Digital Aeronautical Communications System
LME	LDACS Management Entity
LUT	Look Up Table
MAC	Medium Access Control
MAKE	Mutual Authentication and Key Establishment
MF	Multi Frame
MICONAV	Migration towards Integrated COM/NAV Avionics
MIMO	Multiple-Input Multiple-Output
Mod	Modulation
MPSoC	Multiprocessor System-on-Chip
OFDM	Orthogonal Frequency Division Multiplexing
OCXO	Oven-Controlled Crystal Oscillator
PAV	Power Adaptation Value
PDU	Packet Data Unit
PER	Packet Error Rate
PL	Programmable Logic
PS	Processing System
QoS	Quality of Service
RA	Random Access
RACH	Random-Access Channel
RF	Radio Frequency
RFC	Request For Comments
RL	Reverse Link
RS	Reed-Solomon
RTL	Register Transfer Level
RX	Receiving
SAC	Sub-net Access Code
SESAR	Single European Sky Air Traffic Management Research
SDR	Software Defined Radio
SDU	Service Data Unit
SF	Super Frame
SG	Signal Generator
SIB	System Identification Broadcast
SNP	Sub-Network Protocol
SNR	Signal-to-Noise Ratio
SWIM	System Wide Information Management
Sync	Synchronization
TAV	Time Advance Value
TDM	Time Division Multiplex
TBO	Trajectory-Based Operations
TX	Transmitting
UA	Unique Address
USRP	Universal Software Radio Peripheral
VDL Mode 2	VHF Data Link Mode 2
VHF	Very High Frequency
VI	Voice Interface
WAM	Wide Area Multilateration

ACKNOWLEDGMENT

A part of this work was conducted with the overseas research program by the Electronic Navigation Research Institute (ENRI), Japan in 2022. Besides my co-authors from DLR, the authors would like to thank all the members (Dr. A. Filip-Dhaubhadel, M. A. Bellido-Manganell, A. Gürbüz, D. M. Mielke, D. Becker, L. M. Schalk, T. Ewert) of the Aeronautical Communications Group of the German Aerospace Center (DLR), Germany for supporting this cooperative research work. The authors would like to thank the micro/millimeter wave application group members and the former LDACS project members (Dr. Y. Sumiya, J. Kitaori, Dr. S. Futatsumori) at ENRI for supporting the development of the LDACS prototype.

REFERENCES

[1] IATA. "Air passenger market analysis-January 2023." [Online]. Available: Accessed: Apr. 18, 2023. https://www.iata.org/en/iata-repository/publications/economic-reports/air-passenger-market-analysis—january-2023/

[2] ICAO. "Global TBO concept (VERSION 0.11)." [Online]. Available: Accessed: Apr. 18, 2023. https://www.icao.int/airnavigation/tbo/Pages/Why-Global-TBO-Concept.aspx

[3] M. Schnell, U. Epple, D. Shutin, and N. Schneckenburger, "LDACS: future aeronautical communications for air-traffic management," *IEEE Commun. Mag.*, vol. 52, no. 5, pp. 104–110, May 2014, doi: 10.1109/MCOM.2014.6815900.

[4] M. A. Bellido-Manganell et al., "LDACS flight trials: Demonstration and performance analysis of the future aeronautical communications system," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 58, no. 1, pp. 615–634, Feb. 2022, doi: 10.1109/TAES.2021.3111722.

[5] ICAO. "Doc 10039: Manual on system wide information management (SWIM) concept (Draft)." Accessed: Apr. 18, 2023. [Online]. Available: https://www.icao.int/airnavigation/IMP/Documents/SW%20Concept%20%20Dra%20wi%20DISCLAIMER.pdf

[6] Y. J. Morton, F. Diggelen, J. J. Spilker, and B. W. Parkinson, "Ground-based augmentation system," *Position, Navigation, and Timing Technologies in the 21st Century: Integrated Satellite Navigation, Sensor Systems, and Civil Applications*. New York, NY, USA: Wiley, 2021, pp. 259–276, doi: 10.1002/9781119458449.ch12.

[7] S. Han, Z. Gong, W. Meng, C. Li, and X. Gu, "Future alternative positioning, navigation, and timing techniques: A survey," *IEEE Wireless Commun.*, vol. 23, no. 6, pp. 154–160, Dec. 2016, doi: 10.1109/MWC.2016.1500181RP.

[8] N. Mäurer, T. Gräupl, and C. Schmitt, "L-band digital aeronautical communications system (LDACS)," IETF, Fremont, CA, USA, RFC 9372, Mar. 2023. Accessed: Apr. 18, 2023. [Online]. Available: https://www.rfc-editor.org/info/rfc9372

[9] C. Rihacek et al., "L-band digital aeronautical communications system (LDACS) activities in SESAR2020," in *Proc. Integr. Commun. Navigat. Surveill. Conf. (ICNS)*, Herndon, VA, USA, Apr. 2018, pp. 1–8, doi: 10.1109/ICNSURV.2018.8384880.

[10] C. Rihacek, M. Sajatovic, J. Meser, and T. Gräupl, "L-band digital aeronautical communications system (LDACS)—Technical validations in SESAR2020," in *Proc. Digit. Avionics Syst. Conf. (DASC)*, San Diego, CA, USA, Sep. 2019, doi: 10.1109/DASC43569.2019.9081756.

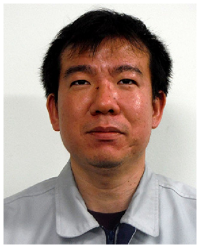
[11] "SESAR2020-PJ14–W2-60-initial LDACS A/G specification, 00.01.00." Dec 2020. Accessed: Jan. 17, 2023. [Online]. Available: https://www.ldacs.com/

[12] T. Gräupl et al., "LDACS flight trials: Demonstration of ATS-B2, IPS, and seamless mobility," in *Proc. Integr. Commun. Navigat. Surveillance Conf. (ICNS)*, Herndon, VA, USA, Apr. 2023, pp. 1–13.

[13] G. Battista, R. Kumar, E. Nossek, and O. Osechas, "Placing LDACS-based ranging sources for robust RNP 1.0 accuracy en-route," in *Proc. Digit. Avionics Syst. Conf. (DASC)*, Petersburg, FL, USA, Sep. 2017, pp. 1–9, doi: 10.1109/DASC.2017.8102135.

[14] M. Felux, T. Gräupl, N. Mäurer, and M. Stanisak, "Transmitting GBAS messages via LDACS," in *Proc. Digit. Avionics Syst. Conf. (DASC)*, London, U.K., Sep. 2018, pp. 1–7, doi: 10.1109/DASC.2018.8569836.

[15] N. Mäurer et al., “Flight trial demonstration of secure GBAS via the L-band digital aeronautical communications system (LDACS),” *IEEE Aerosp. Electron. Syst. Mag.*, vol. 36, no. 4, pp. 8–17, Apr. 2021, doi: 10.1109/MAES.2021.3052318.
[16] A. Filip-Dhaubhadel, M. A. Bellido-Manganell, T. Gräupl, and M. Schnell, “Feasibility assessment of LDACS-based wide area multilateration,” in *Proc. Digit. Avionics Syst. Conf. (DASC)*, Portsmouth, VA, USA, Sep. 2022, pp. 1–10, doi: 10.1109/DASC55683.2022.9925808.
[17] A. Bilzhause, B. Belgacem, M. Mostafa, and T. Gräupl, “Datalink security in the L-band digital aeronautical communications system (LDACS) for air traffic management,” *IEEE Aerosp. Electron. Syst. Mag.*, vol. 32, no. 11, pp. 22–33, Nov. 2017, doi: 10.1109/MAES.2017.160282.
[18] N. Mäurer and A. Bilzhause, “A cybersecurity architecture for the L-band digital aeronautical communications system (LDACS),” in *Proc. Digit. Avionics Syst. Conf. (DASC)*, London, U.K., Sep. 2018, pp. 1–10, doi: 10.1109/DASC.2018.8569878.
[19] N. Mäurer, T. Gräupl, C. Schmitt, G. Dreo-Rodosek, and H. Reiser, “Advancing the security of LDACS,” *IEEE Trans. Netw. Service Manag.*, vol. 19, no. 4, pp. 5237–5251, Dec. 2022, doi: 10.1109/TNSM.2022.3189736.
[20] N. Mäurer, T. Gräupl, and C. Schmitt, “Evaluation of the LDACS cybersecurity implementation,” in *Proc. Digit. Avionics Syst. Conf. (DASC)*, San Diego, CA, USA, Sep. 2019, pp. 1–10, doi: 10.1109/DASC43569.2019.9081786.
[21] N. Mäurer et al., “A secure cell-attachment procedure of LDACS,” in *Proc. Eur. Symp. Security Privacy Workshops (EuroS&PW)*, Vienna, Austria, Sep. 2021, pp. 113–122, doi: 10.1109/EuroSPW54576.2021.00019.
[22] N. Mäurer, T. Ewert, L. J. Jansen, T. Gräupl, K. Morioka, and C. Schmitt, “International LDACS security validation activities a cooperation effort between DLR and ENRI,” in *Proc. Integr. Commun. Navigat. Surveillance Conf. (ICNS)*, Herndon, VA, USA, Apr. 2023, pp. 1–10.
[23] M. Mostafa, M. A. Bellido-Manganell, and T. Gräupl, “Feasibility of cell planning for the L-band digital aeronautical communications system under the constraint of secondary spectrum usage,” *IEEE Trans. Veh. Technol.*, vol. 67, no. 10, pp. 9721–9733, Oct. 2018, doi: 10.1109/TVT.2018.2862829.
[24] M. A. Bellido-Manganell, T. Gräupl, and M. Schnell, “Impact assessment of the L-band digital aeronautical communications system on the joint tactical information distribution system,” *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3629–3641, Apr. 2019, doi: 10.1109/TVT.2019.2898524.
[25] N. Agrawal, S. J. Darak, and F. Bader, “New spectrum efficient reconfigurable filtered-OFDM based L-band digital aeronautical communication system,” *IEEE Trans. Aerosp. Electron. Syst.*, vol. 55, no. 3, pp. 1108–1122, Jun. 2019, doi: 10.1109/TAES.2019.2891092.
[26] S. Roy and A. Chandra, “On the design of variable filtered-OFDM based LDACS for future generation air-to-ground communication system,” *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 69, no. 2, pp. 644–648, Feb. 2022, doi: 10.1109/TCSII.2021.3098913.
[27] D. W. Matolak et al., “Novel filterbank multicarrier waveform for L-band digital aeronautical communications: Initial field test results,” in *Proc. Digit. Avionics Syst. Conf. (DASC)*, San Antonio, TX, USA, Oct. 2021, pp. 1–10, doi: 10.1109/DASC52595.2021.9594295.
[28] M. Keshkar, R. Muthalagu, L. K. Mathew, and A. Rajak, “Deep clipping based interference mitigation technique for LDACS,” in *Proc. Digit. Avionics Syst. Conf. (DASC)*, San Antonio, TX, USA, Oct. 2021, pp. 1–6, doi: 10.1109/DASC52595.2021.9594456.
[29] M. Keshkar, R. Muthalagu, A. Rajak, and L. K. Mathew, “GAE and OBE enhanced interference mitigation techniques in LDACS,” *Aerospace*, vol. 9, no. 45, pp. 1–30, Jan. 2022, doi: 10.3390/aerospace9010045.
[30] “Ettus research$^{\text{TM}}$.” Accessed: Apr. 11, 2023. [Online]. Available: https://www.ettus.com/
[31] N. Agrawal, H. Joshi, S. J. Darak, and F. Bader, “USRP testbed and performance analysis of new reconfigurable LDACS in presence of DME interference,” in *Proc. Int. Symp. Wireless Commun. Syst. (ISWCS)*, Oulu, Finland, Aug. 2019, pp. 400–405.
[32] N. Agrawal, S. J. Darak, and F. Bader, “Spectral coexistence of LDACS and DME: Analysis via hardware software co-design in presence of real channels and RF impairments,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 9, pp. 9837–9848, Sep. 2020, doi: 10.1109/TVT.2020.3002978.
[33] S. Kurz, E. Gringinger, C. Rihacek, and T. Sauter, “High performance implementation of next generation aeronautical communication systems,” *IEEE Open J. Ind. Electron. Soc.*, vol. 3, pp. 700–710, Nov. 2022, doi: 10.1109/OJIES.2022.3220048.
[34] K. Morioka, S. Futatsumori, N. Yonemoto, J. Kitaori, Y. Sumiya, and A. Kohmura, “Rapid prototyping for a future aeronautical mobile communications system using software defined radio,” in *Proc. Integr. Commun. Navigat. Surveillance Conf. (ICNS)*, Dulles, VA, USA, Apr. 2022, pp. 1–9, doi: 10.1109/ICNS54818.2022.9771483.
[35] A. Viterbi, “Error bounds for convolutional codes and an asymptotically optimum decoding algorithm,” *IEEE Trans. Inf. Theory*, vol. 13, no. 2, pp. 260–269, Apr. 1967, doi: 10.1109/TIT.1967.1054010.
[36] H. P. Basavaraj and M. Pappa, “FPGA implementation of high speed convolutional encoder and viterbi decoder for software defined radio,” in *Proc. Int. Conf. Elect. Comput. Commun. Technol. (ICECCT)*, Erode, India, Feb. 2023, pp. 1–4, doi: 10.1109/ICECCT56650.2023.10179840.
[37] “lwIP.” Accessed: May 14, 2023. [Online]. Available: http://savannah.nongnu.org/projects/lwip/
[38] “Wireshark.” Accessed: Apr. 20, 2023. [Online]. Available: https://www.wireshark.org/
[39] Xilinx. “UG1399: Vitis high-level synthesis user guide.” Accessed: Feb. 4, 2023. [Online]. Available: https://docs.xilinx.com/r/en-US/ug1399-vitis-hls
[40] T. Gräupl and N. Mäurer, “An air traffic management data traffic pattern for aeronautical communication system evaluations,” in *Proc. IEEE/AIAA 38th Digit. Avionics Syst. Conf. (DASC)*, San Diego, CA, USA, 2019, pp. 1–6.
[41] L. J. Jansen, T. Gräupl, N. Mäurer, K. Morioka, and C. Schmitt, “A software framework for synthetic aeronautical data traffic generation in support of LDACS evaluation activities,” in *Proc. Integr. Commun. Navigat. Surveillance Conf. (ICNS)*, Herndon, VA, USA, Apr. 2023, pp. 1–11.

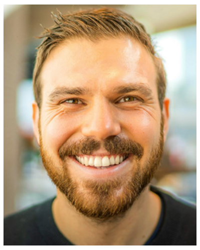


KAZUYUKI MORIOKA (Member, IEEE) received the B.Ec. degree from Hokkaido University in 2005, and the M.E. and Ph.D. degrees in information engineering from Shinshu University in 2009 and 2014, respectively. From 2005 to 2011, he was a Software Engineer with Artiza Networks, Inc. From 2011 to 2012, he was a Technical Engineer with the National Institute of Information and Communication Technology. In 2013, he joined the Electronic Navigation Research Institute, National Institute of Maritime, Port and Aviation Technology, where he is currently a Senior Researcher. In 2022, he was a Visiting Researcher with German Aerospace Center. His research interest is wireless communication systems, including air-to-ground aeronautical communication systems. He is a member of IEICE and IEEJ.

AKIKO KOHMURA (Member, IEEE) received the Ph.D. degree in electronic engineering from the University of Electro-Communications, Tokyo, Japan, in 2007. She is a Chief Researcher with the Electronic Navigation Research Institute (ENRI), National Institute of Maritime, Port and Aviation Technology, Japan. She joined ENRI, and has been working on aeronautical communication, electromagnetic compatibility, millimeter wave antennas, and radars, especially in aviation, including unmanned aircraft issues since 2007. She was a Guest Researcher with the Laboratoire d’Electronique Antennes et Télécommunications, Université Nice Sophia Antipolis, France, from 2011 to 2012.

NARUTO YONEMOTO (Member, IEEE) received the B.E., M.E., and Ph.D. degrees from Saga University, Japan, in 1995, 1997, and 2000, respectively. He joined the Electronic Navigation Research Institute (ENRI), National Institute of Maritime, Port, and Aviation Technology, Japan, where he is currently a Principal Researcher. He was an Visiting Researcher with the Laboratoire d'Electronique Antennes et Télécommunications, Université Nice Sophia Antipolis, France, from 2005 to 2006. He has been also an Associate Professor with the Tokyo University of Marine Science and Technology since 2011. His current research topics are EMC on aeronautics, aeronautical communication systems, and sensing technologies in airports, such as radar and optical imaging.

LEONARDUS J. A. JANSEN received the B.Sc. degree in computing science from Radboud University, Nijmegen, The Netherlands, in 2019, and the M.Sc. degree in cyber security from the University of Twente, Enschede, The Netherlands, in 2022. He works in the domain of cyber security for aeronautical air-to-air communication and LDACS with the Institute of Communications and Navigation, German Aerospace Center, Oberpfaffenhofen.



NILS MÄURER (Member, IEEE) received the B.Sc. degree in computer science from the Technical University Munich, the M.Sc. degree in IT-security and reliability from the University of Passau, and the Ph.D. degree in computer science from the University of the Bundeswehr Munich. He is the Group Leader of the Cybersecurity Architectures Group, Institute of Communications and Navigation, German Aerospace Center. He works on cybersecurity in the field of critical infrastructure, with a dedicated focus on aeronautical communications systems. He is a Systems Architect for the cybersecurity architecture of the L-band Digital Aeronautical Communications System and a lead author LDACS RFC and RFC 9372. His current research topics are symbolic model checker, group key management, lightweight mutual authentication and key establishment procedures, and slim post-quantum cryptography. He won several awards at numerous conferences, such as the Best-of-Conference Award at the Integrated Communications Navigation and Surveillance Conference in 2019.

THOMAS GRÄUPL received the Ph.D. degree in computer science and the M.Sc. degree in mathematics from the University of Salzburg, Austria, in 2011 and 2004, respectively. He is a Senior Researcher with the Institute of Communications and Navigation, German Aerospace Center. He was a Researcher with the University of Salzburg. His current research interests include wireless digital communication systems and the performance evaluation of communications systems through computer simulations. He is an Advisor to the German Panel Member at ICAO and supports the standardization of LDACS in EUROCAE.

MICHAEL SCHNELL (Senior Member, IEEE) received the M.Sc. degree in electrical engineering from University Erlangen-Nuremberg, in 1990, and the Ph.D. degree for his work on wireless communications from the University of Essen (today the University of Duisburg-Essen) in 1997. In 1990, he joined German Aerospace Center (DLR). Since then, he has been working as a Scientific Researcher. He is a Senior Scientist with the Institute of Communications and Navigation, DLR. He is a Lecturer of Multicarrier Communications as well as Aeronautical Communications and acts as a Selected Advisor for the German Air Navigation Service Provider (DFS GmbH) on various committees at EUROCONTROL and ICAO. He has authored/coauthored over 120 publications, including more than 20 journal articles. His main research interests are the development and modernization of CNS technologies for civil aviation and unmanned aerial systems. As Rapporteur of the Project Team "Terrestrial Data Link" within the ICAO Communications Panel, he is organizing the international standardization of the future terrestrial data link L-band Digital Aeronautical Communications System. He is a member of AIAA and VDE/ITG.